NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DTIC
ELECTE
MAR 1 3 1995
S G D

**APPLICATIONS OF DIGITAL VIDEO AND SYNTHETIC
ENVIRONMENTS TO UNMANNED AERIAL VEHICLES**

by

Franklin J.L. Tipton

September 1994

Thesis Advisor:	David R. Pratt
Second Reader:	Michael J. Zyda

19950308 158

REPORT DOCUMENTATION PAGE

Form Approved

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time reviewing instructions, searching existing data sources gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE September 1994	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Applications of Digital Video and Synthetic Environments to Unmanned Aerial Vehicles (U)	5. FUNDING NUMBERS

6. AUTHOR(S) Tipton, Franklin J.L.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION

9. SPONSORING/ MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/ MONITORING

11. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release, distribution is unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The current Army Unmanned Aerial Vehicle (UAV) system has two problems, 1) it does not provide for the direct distribution of live UAV video throughout command posts across their local area networks, and 2) it lacks an automated trainer. To solve the video distribution problem, I studied the UAV system, video compression techniques, and local area network protocols to develop the video distribution model. The approach taken for developing the simulator included researching the operational characteristics of the UAV system and studying the creation of synthetic environments. This thesis develops an architecture for extending the distribution of live UAV video inside the command post using a local area network. It recommends distributing full-motion UAV video over an Asynchronous Transfer Mode (ATM) local area network using the motion Joint Photographic Experts Group (JPEG) compression technique. Additionally, I created an interactive, UAV Simulator using the IRIS Performer applications program interface and C++. The simulator is implemented using two, networked workstations which replicate the functions of the air vehicle and mission payload operators. The workstations communicate across a local area network using the Distributed Interactive Simulation (DIS) protocol.

14. SUBJECT TERMS Autonomous vehicle, Digital Video, Simulations, Synthetic Environments	15. NUMBER OF PAGES 91
	16. PRICE CODE

17. SECURITY CLASSIFICATION Unclassified	18. SECURITY CLASSIFICATION Unclassified	19. SECURITY CLASSIFICATION Unclassified	20. LIMITATION OF ABSTRACT UL

Approved for public release; distribution is unlimited

APPLICATIONS OF DIGITAL VIDEO AND SYNTHETIC ENVIRONMENTS TO UNMANNED AERIAL VEHICLES

by

Franklin J. Tipton
Captain, United States Army
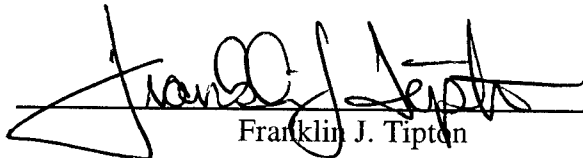*B.S., Georgia State University, 1984*

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1994

Accesion For		
NTIS CRA&I	☒	
DTIC TAB	☐	
Unannounced	☐	
Justification		
By		
Distribution /		
Availability Codes		
Dist	Avail and / or Special	
A-1		

Author: _____
Franklin J. Tipton

Approved By: _____
David R. Pratt, Thesis Advisor

_____
Michael J. Zyda, Second Reader

_____
Ted Lewis, Chairman,
Department of Computer Science

iii

ABSTRACT

The current Army Unmanned Aerial Vehicle (UAV) system has two problems, 1) it does not provide for the direct distribution of live UAV video throughout command posts across their local area networks, and 2) it lacks an automated trainer.

To solve the video distribution problem, I studied the UAV system, video compression techniques, and local area network protocols to develop the video distribution model. The approach taken for developing the simulator included researching the operational characteristics of the UAV system and studying the creation of synthetic environments.

This thesis develops an architecture for extending the distribution of live UAV video inside the command post using a local area network. This architecture specifies bringing UAV video inside the command post via the Joint Surveillance and Target Attack Radar System's (JSTARS) Ground Support Module. Further, it recommends distributing full-motion UAV video over an Asynchronous Transfer Mode (ATM) local area network using the motion Joint Photographic Experts Group (JPEG) compression technique. Additionally, I created an interactive, UAV Simulator using the IRIS Performer applications program interface and C++. The simulator is implemented using two, networked workstations which replicate the functions of the air vehicle and mission payload operators. The workstations communicate across a local area network using the Distributed Interactive Simulation (DIS) protocol.

TABLE OF CONTENTS

x

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. GENERAL DESCRIPTION

This thesis recommends two enhancements to the Unmanned Aerial Vehicle (UAV) system used by the United States Army. First, this thesis recommends an architecture for the distribution of live, full-motion UAV video across a local area network (LAN). Secondly, this thesis produces a UAV simulator for training operators on the UAV system.

B. MOTIVATION

One of the greatest concerns of tactical commanders throughout history has been the lack of the ability to "see over the hill" and determine the disposition of opposing forces. From high altitude balloons used during the Civil War to reconnaissance satellites used during Operation DESERT STORM, commanders have depended on various remote sensors for monitoring the enemy situation. The common problem among all of these sensors has been the time delay in getting the information from the sensor to the decision maker. In other words, the lack of a real-time, tactical imagery asset.

Beginning in fiscal year 1994, the US Army began fielding a system with this potential - Unmanned Aerial Vehicles [BRIE92]. For the first time in history, tactical commanders will have an organic asset capable of providing real-time, tactical imagery to all of the key players within the command post, simultaneously. This thesis addresses two shortcomings of the current UAV system: the video distribution bottleneck and the lack of an automated trainer.

The current system restricts the distribution of live video from the UAV to three entities: the UAV Ground Control Station, the Ground Support Module, or a remote video terminal. This scheme of video distribution does not realize the full potential of the UAV system. As a potential solution, this thesis recommends an architecture that permits the commander, as well as his staff, to simultaneously observe live UAV video.

1

Additionally, this thesis develops a UAV simulator. There is currently no similar training device in use by the Department of Defense's UAV Training Site at Fort Huachuca, Arizona. UAVs cost approximately one million dollars each and all flight training is currently conducted using actual vehicles. The use of a UAV simulator would increase operator proficiency by providing more hands-on training without the wear and tear on actual UAVs.

The remainder of this chapter discusses the origins of the UAV program and provides an overview of the UAV system to include a more detailed description of the shortcomings addressed by this thesis.

1.   The Military Intelligence Relook

In the summer of 1991, a task force "MI Relook" was formed with representatives from: the United States Army Intelligence Center and School, Headquarters Department of the Army, Deputy Chief of Staff for Intelligence, and Deputy Chief of Staff for Operations [BRIE92]. The MI Relook task force conducted a comprehensive review of Army intelligence: missions, forces, equipment, training, doctrine and concepts. They interviewed combat commanders at all levels to determine what commanders expect in terms of intelligence support. As the result of the MI Relook study, six chief functions for Army intelligence were defined. These functions are: indications and warning, intelligence preparation of the battlefield, protect the force, situation development, target development, targeting, and battle damage assessments. Also, the MI Relook task force developed an intelligence architecture to accomplish these chief functions. This architecture emphasized the following systems: Unmanned Aerial Vehicles, Joint Surveillance Targeting And Reconnaissance System (JSTARS), Ground Support Module (GSM), Secondary Imagery Dissemination, All -Source Analysis Section (ASAS). While a detailed description of most of the above systems is beyond the classification level and scope of the scope of this thesis, this thesis focuses on the UAV system. Oddly enough, more information on the other

systems is available in [JANE93]. The goal of the architecture defined by the MI Relook is to "push" intelligence directly to the commander while simultaneously giving the commander the ability to "pull" additional intelligence needs from the system.

2.   Unmanned Aerial Vehicle Overview

The first of the Army's UAV units will be fielded during fiscal year 1994. Under the current plan, Army units, brigade and higher, will have an organic UAV organization. The UAV system consists of remotely controlled aircraft and their Ground Control Stations. Once launched, the Ground Control Station maintains radio contact with the UAV to provide command and control of the UAV aircraft. The UAV provides analog sensor information to the GCS using radio. The Ground Control Station is manned by intelligence analysts who generate textual reports based on significant events detected by the UAV's sensors. The UAV can carry one of two sensors on board, a black-and-white video camera or an infra-red camera. The Ground Control Station receives and records UAV video and infra-red mission results on a video-cassette recorder.

3.   Description of Problem Areas

As stated above, this thesis recommends two enhancements to the UAV system. The first enhancement describes an architecture to support the distribution of full-motion video throughout the command post. The second enhancement develops a simulator for the UAV to assist in the training of new operators. This section first describes the problem with video distribution then describes the utility of a UAV simulator.

a.   UAV Video

This section describes the bottlenecks associated with UAV video distribution. Under the present system, the video feed from an UAV terminates at one of three destinations: the Ground Control Station (GCS), the Ground Support Module (GSM), or a remote video terminal. The GCS is the UAV's controlling entity and will normally be

physically removed from the supported commander. This configuration does not permit the supported commander, to see live video. Instead, the commander must rely on textual reports generated by analysts at the GCS. The timeliness of this method further suffers from the time lag involved in the creation and distribution of this textual information over a potentially congested communications network.

There is another system that is capable of receiving live UAV video. This system is the GSM and is co-located with the brigade TOC. The GSM is capable of receiving information from a number of systems providing the GSM is within the sensor's downlink. The GSM's ability to receive UAV video is a secondary mission. The GSM will permit the commander to observe the live UAV video feed provided the commander is physically located within the GSM; however, this is not a practical solution.

Remote Video Terminals (RVT) are another device used for displaying live UAV video. An RVT is essentially a television where personnel can gather around at mission time to watch UAV video. The RVT may be physically located within the TOC and allow the commander and the staff to watch live UAV video, but, in order to do so, they must leave their work areas to congregate in front of the screen. There are also a limited number of RVTs available to each UAV organization.

Unless a commander is physically located within the GCS, or the GSM, or has access to an RVT, he/she cannot see live UAV video in real time. This does not realize the potential of the UAV. Ideally, there should be the capability to view the live UAV video simultaneously by all key personnel within the command post without having to leave their places of duty. This could be accomplished by implementing a LAN within the command post capable of processing full motion video. This would allow key members of the command post to view and, most importantly, react to the live UAV video from their respective battle stations.

The integration of UAV video across a LAN within the TOC would permit the following scenario. The UAV sensor detects an enemy unit moving into unprepared positions. The commander and his staff detect and observe this movement. The commander determines that the enemy force is within artillery range and orders his fire support officer to engage. The fire support officer extracts targeting information from the UAV telemetry data and calls for fire. In real time, the commander and his staff can observe the engagement, adjust fire, and render battle damage assessments - all in real time. Of the six chief functions of intelligence defined by the MI Relook, this scenario satisfies four of the six: indications and warning, situation development, targeting, and battle damage assessment. This capability can not exist without the ability to distribute live, full-motion UAV video throughout the tactical command post.

b. UAV Simulator

Simulators are commonly used to train operators and to test employment concepts for various systems. The same benefits can be obtained with the use of a UAV simulator. UAVs cost approximately one million dollars each. Because there is presently no simulator available for training new operators on the UAV system, all operators are trained using actual vehicles. A simulator could be used to train new operators how to fly the UAV, how to control the payload, as well as train the operators how to more effectively interact for mission accomplishment. Additionally, since UAVs are relatively new to the inventory, a simulator could be used to more fully develop doctrine for the employment of UAVs.

A UAV simulator could be incorporated into existing combat simulations such as the Naval Postgraduate School Networked Vehicle Simulator (NPSNET) [MZPB94], where the UAV simulator could be used to conduct reconnaissance. Currently in NPSNET, the relative location of all entities inside the virtual world are indicated using a heads up display. The placement of icons on the heads-up display provides the user with

an omniscient perspective on the relative location of all players inside of the virtual world. By only telling the users the location of entities detected by fellow players or what the UAV has detected, this more accurately reflects the way information is gathered in actual combat conditions. The integration of UAV play into combat simulations such as NPSNET provides a low cost alternative to learning how to incorporate UAVs into the current force structure when compared to using actual equipment during field training exercises. For example, prior to a rotation to the National Training Center (NTC), the participating unit could train on a combat simulator such as NPSNET using the Fort Irwin terrain data base. The participating armor, aviation, and intelligence organizations could conduct combined arms training for their soldiers and learn how to incorporate the use of UAVs into their tactical standard operating procedures.

C.   OBJECTIVES

The objectives of this study are to recommend an architecture for distributing full motion, UAV video across the TOC's local area network. Additionally, this thesis develops a UAV simulator designed to emulate the flight of an UAV through a three-dimensional, virtual world by imitating the functions of the UAV pilot and payload operator.

D.   SCOPE, LIMITATIONS, AND ASSUMPTIONS

This study assumes that a coaxial cable is established between the GSM and the tactical operations center. The purpose of this coaxial cable is to provide a conduit for live analog video to serve as an input to the LAN located within the TOC. The GSM is produced by Motorola and is a proprietary system.

For the simulator, sufficient hardware and software is available to produce a UAV simulator. However, the resulting application will be hardware-specific in that it will require a Silicon Graphics' Reality Engine and the Performer applications program

6

interface. This particular simulator runs on the IRIX operating system 5.2 and Performer version 1.2.

E. SUMMARY OF CHAPTERS

This thesis is organized into two logical parts. Part one, which consists of Chapter II, discusses the technical issues involved in the implementation of the video distribution model. It discusses the video signal, video compression techniques, considerations for networking full motion video as well as other implementation issues. Part two, which is essentially the remaining chapters, describes the UAV simulator. The breakdown for part two is as follows. Chapter III describes the UAV Simulator as well as design considerations. Since the simulator is comprised of two modules each module is discussed separately. Chapters IV describes the Air Vehicle Operator module. Chapter V describes the Mission Payload Operator module. The conclusions of this thesis are presented in Chapter VI.

8

II. NETWORKING LIVE VIDEO

This chapter describes an architecture that allows networked workstations within a tactical operations center (TOC) to receive, display, process, and store live, full-motion video from an UAV. We begin with a description of the system requirements and then introduce the video distribution model. This model provides the framework within which the technical issues related to networking live video are discussed throughout the remainder of this chapter.

A. SYSTEM REQUIREMENTS

This section describes the system requirements used to develop the video distribution model. These requirements are based on what should be possible in the near future and are not driven soley by the current state-of-the-art. Because much of the technology required to implement the video distribution model parallels what is happening with the digitization of the television industry and the development of video on demand services, these technologies will be economically available in the near future. In order to implement a specific system, trade-offs based on price-performance analysis must be made and are beyond the scope of this thesis.

The video distribution model is designed to permit live, full-motion video from a UAV to be multicast to all subscribers on the LAN within the TOC. Further, the model provides sufficient bandwidth for all users to view two simultaneous video feeds: one live and one previously recorded video segment. These video sources could be, for example, the live feed from a UAV as well as the playback of a previously recorded video segment. In addition to the network providing sufficient bandwidth to simultaneously process multiple video streams, sufficient bandwidth remains to enable the coexistence of traditional file server without degraded performance. This capability would permit traditional LAN services such as resource sharing, electronic mail, and file transfers to exist along with the capability to network full-motion video. Lastly, the model must provide a mechanism for

the recording, cataloging and playback of UAV video segments. These video processing capabilities must be automated and as user-friendly as possible.

B. THE VIDEO DISTRIBUTION MODEL

Figure 1 illustrates the video distribution model. The model describes the distribution of video from a deployed UAV to the users within the TOC. The deployed UAV transmits information to its controlling Ground Control Station using a radio link. Assuming that the Ground Support Module (GSM) is within the UAV's telemetry downlink range, it is also capable of receiving the analog video feed from the UAV. This model assumes that there exists a conduit to facilitate the transport of UAV video from the GSM to the network within the tactical operations center. This is represented graphically by the line running from the GSM to the S-2's workstation. It is at the S-2's workstation, also functioning as the video front end, where the scope of this recommended architecture begins.

At this video front end is a coaxial cable carrying an analog, NTSC video signal. In order to process this analog video inside a computer, it must first be converted into a digital format. Because digital video is so demanding in terms of bandwidth and storage requirements it must first be reduced, or compressed, before computers and networks can efficiently process the digital video. Once the video has been digitized and compressed, it can be manipulated by computers and transported by a computer network more efficiently. Next, the digital video must be multicast across the network. The ability to multicast allows all network subscribers to have simultaneous access to UAV video. Multicast establishes a one-to-many relationship where users can share one video stream on the network. Additionally, the video must be recorded onto some medium. This permits the playback and analysis of UAV video for analysis. Video storage is accomplished by the video server in this architecture. This will allow users to request video segments that have been recently recorded. Lastly, the network can not be overloaded by video processing alone, it must also be capable of performing traditional network functions.
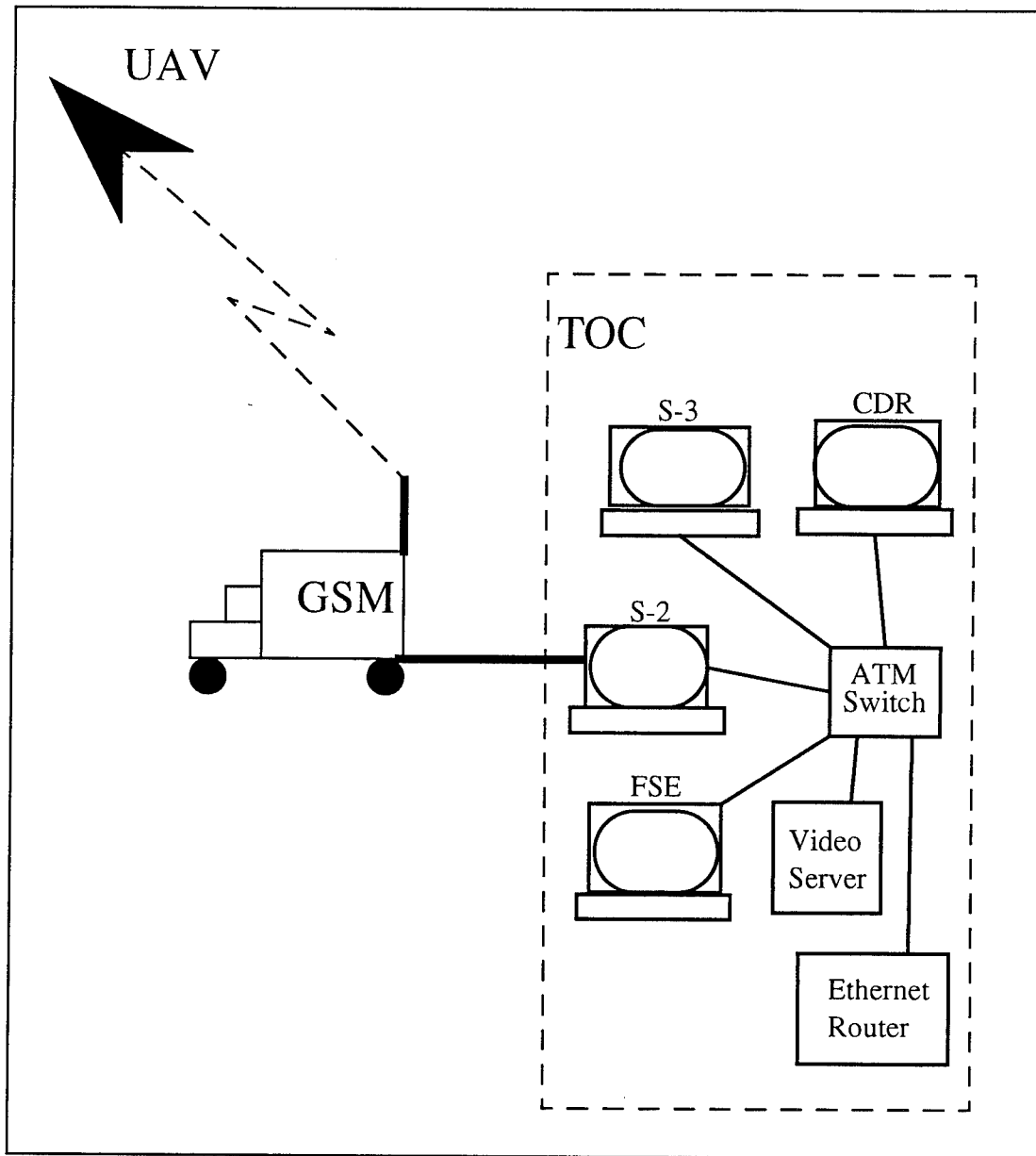
Figure 1: The Video Distribution Model

The video distribution model identifies the key technical issues involved with processing the live video across a local area network. These key issues are: analog/digital conversion, compression, networking, storage of video, interoperability. The remainder of this chapter discusses these issues separately.

C.   ANALOG/DIGITAL CONVERSION

The first issue relates to digitizing the incoming UAV video signal. The key concept to understand is that digitized video demands tremendous bandwidth and storage requirements. One second of uncompressed, black-and-white full Source Interchange Format NTSC video requires 86 million bits to represent those images digitally. This becomes significant once you consider that the prevailing local area network protocols, Ethernet and FDDI, have theoretical data rate maximums of 10 and 100 megabits per second, respectively. This 86 million bits per second was calculated using an NTSC, D-2 quality video oversampled at three times with a subcarrier frequency of 3.58 megahertz, quantized at eight bits per sample [INGL93]. The equation for this calculation is 3.58 (subcarrier frequency) x 3 (sampling rate) x 8 (bits for quantizing analog to digital) = 85.9 megabits per second. The Hunter UAV has a maximum endurance of 12 hours [JANE93]. This equates to almost four terabytes of storage required to store all 12 hours of uncompressed video. This illustrates the problem of processing and storing UAV video. These extreme processing/storage requirements must be reduced in order to make the system tractable.

D.   COMPRESSION

This section provides an overview of video compression technology. The choice of a compression technique is a major design consideration for systems that process digital video. The choice of a compression technique dictates the network topology, bandwidth requirements, and storage requirements for the video server as well as hardware

requirements for each attached workstation. While each technique achieves the goal of compression, each technique uses a different approach - all of which have their own strengths and weaknesses. This section provides a brief overview of compression techniques. The goal of the section is to explain why this thesis recommends Motion JPEG (Joint Photographic Expert Group) as the compression standard for the video distribution model.

1.   Compression Categories

Compressing the video signal is necessary in order to reduce storage and bandwidth requirements. There are two general categories of compression algorithms - lossless and lossy. Lossless compression applications generally offer low compression ratios and reduce the size of video files within a range from 1.5:1 to 4:1 [GUGL93]. Using the 86 megabit per second baseline figure, lossless compression can reduce the storage requirement for one second of video to approximately 21 megabits per second. Because of the limited amount of compression obtained, most compression algorithms are lossy. Lossy compression algorithms lose more information during compression but achieve higher compression ratios. The higher the compression ratio, the more image data that is irrevocably lost.

Not only are compression algorithms described in terms of lossless and lossy but they are further categorized in terms of intraframe and interframe encoders. These distinctions describe the compression method and how it compresses a series of frames. Intraframe encoders use only the information within an individual frame to perform compression of the image. In order to achieve compression between frames, where redundant information such as a static background is discarded, interframe techniques are employed. There are also methods which use a combination of interframe and intraframe encoding techniques.

Another trade-off used in enhancing compression is to manipulate the source interchange format (SIF). The UAV electro-optical payload is a black-and-white NTSC camera with a SIF of 640 x 480 pixels. By cutting the SIF in half, to 320 x 240,the effective bandwidth requirements for digital video are also reduced by two-thirds from 307,200 to 76,800. Generally speaking, existing compression techniques reduce the SIF anywhere from one-half to one-sixteenth. But in doing so, the corresponding picture on the computer display at playback is reduced in size and quality thus reducing the information available within a given frame of information.

The remainder of this section describes JPEG, motion JPEG, MPEG-1, and MPEG-2 compression techniques to illustrate their differences. These discussions provide an overview of how the techniques work, describe their basic features in terms of SIF formats as well as their relative strengths and weaknesses with respect to the video distribution model.

2. JPEG

JPEG stands for Joint Photographic Experts Group - the name of the group that wrote the standard. JPEG is a lossy, full SIF, intraframe compression algorithm. It was designed for compressing either full-color or gray-scale photographs. JPEG exploits limitations of the human eye. Small color details are not perceived as well as small details of contrast by the human eye. JPEG compresses color information more heavily than luminance. JPEG's compression can be varied by adjusting compression parameters thus allowing the image maker to trade-off file size against image quality. Optimally, JPEG can achieve 10:1 to 20:1 compression ratios. However, gray-scale images, such as black-and-white UAV video, do not compress by these large factors because the human eye is very sensitive to variations in brightness. Therefore, the threshold of visible loss for gray-scale is around 5:1 [GAIL93]. Once again referring to the 86 megabit baseline, JPEG can only reduce bandwidth and storage requirements to a maximum of 17.2 megabits per second.

14

3. Motion JPEG

JPEG was designed for still photographs. In order for JPEG to be extended to full-motion video, motion JPEG was developed. Motion JPEG is a full-SIF compression technique in which a video stream is digitized and compressed as a series of independent frames. Since each frame is independent, motion JPEG allows for slow-motion replay as well as frame-by-frame analysis. These capabilities are important for post mission analysis of UAV video.

Motion JPEG is symmetrical with respect to compression hardware meaning that it requires a special board to be installed on each workstation. It requires a card for digitization and compression of the incoming analog signal as well as another card for playback of full-motion, digital video for each attached workstation.

4. MPEG-1

The Motion Pictures Expert Group (MPEG) algorithm was designed to accommodate motion video. It uses a standard resolution of 320 by 240 pixels for full motion video. MPEG defines a bit stream for compressed video and audio optimized to fit into a bandwidth of 1.5 megabits per second. MPEG implements both intraframe and interframe encoding techniques. It uses intraframe encoding to remove redundancies within individual frames and uses interframe coding to reduce redundancies between frames. For example, if the background of a video stream remains constant from frame to frame, MPEG will save the background once and store only the differences between frames. After intraframe encoding, MPEG analyzes two subsequent images and uses movement compensation. It determines for each pixel the prediction model (inter and intraframe) and transmits the quantized differences as the prediction model changes. MPEG has 3 frame formats: Image, Predicted, and Bi-directional. Image frames are complete bit images of a single frame and are only sent periodically in order to correct the transmitted image. Predicted frames update an image from the last image frame or

15

predicted frame. Bi-directional frames describe the difference from the last or next high quality image [GAIL93]. The ratio of B and P frames is configurable. MPEG supports compression rates of up to 50:1 and is asymmetrical with respect to compression hardware. It requires hardware for compression but can decompress in software.

The major disadvantage with using MPEG for intelligence purposes is that, for optimal compression, MPEG assumes a moving object against a stable background. Typical UAV video segments will include a moving background. This reduces the potential of the interframe encoding capability of MPEG. Additionally, UAV video is produced for intelligence purposes and the higher the fidelity, the greater the information potential of the image. Using P and B frames, MPEG predicts what the image should be. Since UAV video will be used for reconnaissance, targeting, and battle damage assessments there must exist the capability to analyze each individual frame. Lastly, MPEG takes NTSC from its base SIF, 640 x 480, and reduces it to 320 x 240. Information will be lost as MPEG "shrinks" the base format to half its original size. Also, watching networked video at less than full SIF is less desirable than watching the video at full NTSC SIF.

5. MPEG-2

While MPEG-1 was designed to equal the quality of a VHS video tape, MPEG 2 is an emerging standard designed for broadcast quality video. The emerging MPEG-2 standard defines a bandwidth of 4-9 megabits per second. MPEG- 2 defines a SIF format of 704 x 576 pixels - more than four times the pixel count of MPEG 1[FOGG94]. Like MPEG-1 it uses inter and intraframe encoding. MPEG-2 has been chosen as the compression algorithm for high definition television and supports full NTSC SIF.

MPEG-2 is disadvantaged by its implicit lack of support for frame-by-frame, slow motion replay as well as freeze frame analysis. Neither MPEG standard defines this functionality and if available would be vendor specific.

E. NETWORKING ISSUES

This section recommends Asynchronous Transfer Mode (ATM) as the appropriate Local Area Network (LAN) protocol for distributing full motion video within the TOC. Once the UAV video has been digitized and compressed, it must be sent out over the network for all the users within the TOC. Networking full motion video requires an isochronous capability of the network. When a video segment is ready to be sent out over the network each stream must be guaranteed access to the network within stringent time parameters, with the bandwidth required [PART94]. For successful delivery of full motion video, the service requirements for isochronous media must be satisfied. The use of motion JPEG for video compression, defines a minimum bandwidth of 20 megabits per second, per video stream. The next task is to chose a network protocol that supports the distribution of multiple isochronous video segments. This section discusses the issues leading to recommending ATM for implementing the video distribution model. First, this section provides a brief overview of networking issues then discusses three, commercially-available, networking protocols: Ethernet, Fiber Distributed Data Interface (FDDI), and Asynchronous Transfer Mode (ATM) with respect to their implementation in the video distribution model.

1. Issue Overview

Once the compression technique is defined, the next problem is to choose a network protocol that is capable of processing multiple, full-motion video segments. The combination of improved workstation performance, improvements in fiber technology, and the introduction of multimedia are challenging the traditional networking infrastructure. Future network designs will provide for both data and interactive multimedia services. The first generation networks, Ethernet and Token Ring, focused on providing data services. Second generation networks like FDDI simply improved the ability to provide data services. Future networks must support isochronous media such as voice and video.

17

Isochronous traffic has regular time intervals between samples as they are sent into the network. Once sample "n" is played, sample "n+1" must be played within a stringent, fixed-time interval. For example, to avoid flickering video, frames must be displayed at fixed intervals apart at a minimum rate of 30 frames per second. Additionally, all of the data for a given frame must fit into a fixed amount of bandwidth. The key consideration here is fixed versus variable rate video channels. The remainder of this section will briefly discuss Ethernet, FDDI, and ATM and these networking issues as they relate to the video distribution model.

2. Ethernet

Ethernet is a bus-based topology that defines a ten megabit per second maximum data rate [STAL91]. Subscribers to the network gain access using the Carrier Sense Media Access/Collision Detection (CSMA/CD) protocol. CSMA/CD and its limited bandwidth are the problems associated with processing full-motion video over ethernet. CSMA/CD allows media access on a first-come, first-serve basis. On the TOC LAN there will be many users and the potential for significant network activity. CSMA/CD provides no guaranteed bandwidth for any user; therefore, when UAV video becomes available for the network, it would only be guaranteed delivery if the network was not busy doing anything else. This lack of guaranteed, dedicated bandwidth would interrupt the distribution of full motion UAV video. The second restriction of Ethernet is its limited bandwidth. Using motion JPEG precludes the implementation of the video distribution model on Ethernet. The ten megabit per second theoretical maximum of ethernet is not sufficient for motion JPEG's twenty megabit per second minimum requirement.

3. FDDI

The Fiber Distributed Data Interface (FDDI) protocol is defined by a dual-ring topology that is rated at 100 megabits per second. The network is configured such that there

is a Target Token Rotation Time (TTRT) within which the network keeps tokens rotating around the network. The TTRT is the summation of each station's synchronous and asynchronous allotment, network propagation delay, and each attached station's processing time. In the FDDI protocol, each network subscriber is allotted a synchronous and asynchronous time slice within which each station may transmit over the network. Synchronous bandwidth refers to guaranteed time slots within which a station can transmit. The asynchronous portion refers to what can be transmitted provided the network is not behind [STAL91]. However, most implementations of FDDI do not define a synchronous time slice for each attached workstation. In early implementations of FDDI, vendors emphasized data services and packet transmission and defined only asynchronous time slots for attached stations. Real-time, interactive services, however, need a fixed connection that is not interrupted by packet data services - a synchronous allotment. To get around this problem, FDDI II was introduced. FDDI II implements the synchronous portion of the FDDI standard which guarantees a fixed amount of bandwidth per TTRT for each station.

Assuming that each station's synchronous allotment is configurable in FDDI II, it would be necessary to define a synchronous allotment consistent with the maximum bandwidth required for each station that transmits digital video on the network. The first problem with this approach is that the network administrator must define this synchronous allotment. Assuming that we implement the video distribution model using motion JPEG over FDDI II, video transmission must be performed using fixed bit rate channels. Using the aforementioned motion JPEG compression rates, the bandwidth required for a single frame can vary between 86 and 20 megabits per second. The problem is defining a synchronous allotment for a given station when the input is of variable length. If the synchronous allotment is defined too high this results in wasted bandwidth. On the other hand, if it is set too low, information will be lost. The problem is exacerbated when

19

contemplating multiple video streams over the network and determining whether or not the network protocol is capable of handling multiple motion JPEG video streams. If optimal compression is achieved, then FDDI II can handle up to four simultaneous motion JPEG video streams and still have 10 - 15 percent of bandwidth available for asynchronous traffic. However, the variable rate nature of motion JPEG compression negates this as a viable alternative.

4. ATM

The argument for FDDI and defining synchronous allocations for network entities illustrates the problems associated with using an older protocol for the newer isochronous media such as voice and video. Asynchronous Transfer Mode (ATM) is commonly described in literature as the network protocol of the future. ATM was designed from its conception to be scalable and handle isochronous media. This section provides a brief overview of the ATM protocol as it relates to the video distribution model.

ATM is the underlying transmission system for implementing the Broadband Integrated Services Digital Network (B-ISDN). It is capable of providing a wide range of communications services over a wide range of bit rates. Current ATM standards allow network users to access the ATM network at speeds ranging from 51.4 to 622 megabits per second and it is expected that gigabit speeds will also be supported [PART94].

An ATM network is implemented using a star topology with the central switching fabric being used to provide connectivity between the network's attached stations. The physical media that provides this connectivity is defined by the Synchronous Optical Network (SONET) protocol. SONET is an international standard and prescribes data rates in increments of 51.4 megabits per second using the Optical Carrier (OC) designation. For example an OC-1 SONET link is capable of 51.4 megabits per second where an OC-3 rated media can support three times that data rate or approximately 622 megabits per second.

There are OC ratings currently defined up to level 48 which are rated at 2,488.32 megabits per second.

In order to support a wide range of communications services, ATM defines four adaptation layers (AALs). The purpose of these AALs is to package differing data types such as data, voice, and video into a series of cells that can be sent out over ATM connections and be restructured in the proper format at the destination [PART94]. AAL1 supports continuous bit-rate applications such as an MPEG-1 video stream. MPEG-1 defines a 1.5 megabit per second video stream and has timing requirements between frames. AAL2 supports variable bit-rate applications with timing requirements. As an example, AAL2 would support motion JPEG video streams. ATM differs from FDDI with respect to implementation in the video distribution model in that ATM would dynamically allocate the necessary bandwidth for a video frame where, in FDDI, these are statically defined. This provides for the most efficient method for the allocation of bandwidth for a video application. Assuming at least 20 per cent compression for two simultaneous motion JPEG video streams, ATM could support these JPEG video streams in addition to a moderate amount of file services. AAL 3/4 supports connection-oriented applications such as the file transfer protocol where first a connection between communicating entities must be established then the bytes of information arrive in order. AAL5 supports data communications that employ datagrams. This method does not require a connection to be established and maintained but use a store and forward approach for routing information through a network.

Because of its built-in ability to support variable bit-rate applications and incrementally support higher data rates, ATM best satisfies the network protocol requirements for implementing the video distribution model. ATM is the only available protocol that supports variable-bit rate video by dynamically allocating bandwidth based on the application's requirement. This optimizes the allocation network bandwidth. Also,

because the physical media is based on the internationally accepted SONET standard, ATM provides a migration path to interoperability as well a path to gigabit networking speeds. This allows for future growth using the infrastructure of the existing ATM network.

5. Video Server

This section describes the key issues concerned with implementing a video server in the video distribution model. The key considerations are cataloging of video clips, thermal recalibration of the heads on the video server, and the migration/archiving of information. This section discusses these issues.

The first consideration is to develop a method for annotating the UAV video. Each video segment must be uniquely identified in order to manage video segments. Along with video, the UAV's telemetry feed includes key UAV parameters such as: heading, altitude, time, and field of view. Perhaps this information can be stripped from the video feed by the system and be provided digitally to the video server to assist in the cataloging of video segments. The point is that here must be some method devised for cataloging video segments - either using the telemetry data from the UAV or by using some user supplied information, or a combination of both. Once the system for cataloging video segments has been established, the next consideration is the technique used by the system for periodic recalibration.

For traditional file servers, disk access consists of fetching and writing blocks of data. These blocks of data are relatively small when compared to video files. Video segments will normally be orders of magnitude larger. Getting disk drives which were originally designed for short bursts of information to handle much larger video files introduces the problem of thermal recalibration. As the temperature of a spinning disk changes so does its physical dimensions. To keep the heads calibrated, the device conducts periodic recalibration. The interval between this periodic recalibration becomes an issue when the system decides to recalibrate in the middle of transferring a large video file

[LEHT94]. The handling of thermal recalibration by a server is one reason why a specialized server is required for video.

Another key feature of a video server is its ability to automate the migration of video segments as they are no longer needed. Based upon a least recently used (LRU) algorithm, the video server could control the migration of older video segments as the disk begins to reach capacity. The algorithm simply prescribes that the oldest video segment, or the one that has not been used most recently, is identified and is automatically backed up onto some other storage medium. This archiving medium could be tape or another disk system. The algorithm calls for the backup and removal of enough of the older video segments until there is sufficient room for the new video segment. This automatic migration capability would free the user from having to perform the same function manually.

6. Interoperability

This section discusses the final issue in implementing the video distribution model. Interoperabiliity has two aspects, internal and external. Internal interoperability concerns itself with the ability to incorporate digital video in an distributed environment and continue to meet the needs of all users. External interoperability concerns itself with the ability to share digital video with other command posts.

a. *Internal Interoperability*

Internal interoperability is a key consideration because there is currently no standard local area network protocol used in tactical operations centers. The TOC is comprised of several heterogeneous functions ranging from administration and logistics to fire support and civil affairs. Each function has its own "stove-pipe" support organization. Automation tools in support of each staff or support function are also traditionally developed vertically. The efficient horizontal integration of these functions within the TOC

23

improves combat effectiveness. The intent of the video distribution model is to provide a robust and upgradeable architecture to facilitate the horizontal integration of these vertically developed systems.

Implementing the video distribution model using ATM provides this robust and upgradable architecture. Since ATM is designed to support all media, including voice and video, then it is feasible that ATM can support present and future applications necessary for supporting the needs of the users in the TOC. Also, since the ATM protocol provides a migration path to higher data rates, it provides the backbone necessary for future growth as well.

The inclusion of an ethernet router into the video distribution model provides for backward compatibility for applications that were designed using the Ethernet protocol. The Ethernet router functions as a translator that converts ethernet protocols to ATM and vice-versa.

b. External Interoperability

The external interoperability issue concerns itself with the ability for a given command post to share digital video with other command posts. For example, the ability for a brigade command post to share timely digital video with its subordinate battalions or with its next higher, the division headquarters.

The high data rates necessary for digital video are possible because of LAN technology. Digital information is transmitted across some shared or switched media such as copper wires or fiber optic cables. These media provide the necessary bandwidth. The same high data rates are not sustainable using the Army's current generation of wireless communications systems. Table 1 lists the military's current radio suite and the data rates obtainable [EVAN94].

System	Maximum Data Rate (kbps)
EPLRS and TACFIRE	1.2
AN/VSC-7 tactical satellite communications link	2.4
SINCGARS	16.0
Mobile Subscriber Equipment Terminal	32.0
JTIDS	115.0
Satellite-communications link on Navy Ships	176.0
ATCCS	256.0
AN/TSC-85/93 satellite communication link	1544.0
TRC-170 strategic radio	4608.0

TABLE 1: Digital Data Rates For Military Systems

The primary tactical radios used by Army units are depicted in the first four entries in Table 1. The Mobile Subscriber Equipment (MSE) system is the Army's most modern communications system and is bandwidth limited to a maximum data rate of 32 kilobits per second. None of these systems are capable of transmitting full-motion digital video using the motion JPEG compression technique. Only the last two entries are capable carrying an MPEG-1 compressed digital video stream; however, these systems are not typically deployed with tactical units.

F. SUMMARY

This chapter specifies an architecture to support the distribution of full-motion video throughout the TOC. It specifies the use of the motion JPEG compression technique primarily because of the need to perform frame-by-frame playback for analysis. Because of the bandwidth required for implementing motion JPEG, the video distribution model

25

specifies the use of ATM for its LAN protocol. The ATM standard supports multiple media types to include voice and video as well as higher data rates than FDDI or Ethernet. The ATM protocol also specifies a migration path for even higher data rates - potentially into the gigabit range. The model also prescribes the use of a video server for storage, retrieval, and migration of video segments. This feature makes the system more user friendly as the video server will control the migration of older video segments.

The capability exists to distribute digital video over a local area network. There is another problem associated with trying to distribute that same full-motion digital video stream to other tactical operations centers across the battlefield. The bottleneck here is the limited data rate sustainable by the current inventory of communications systems. Either these systems must find a way to obtain higher data rates or there must be some way to gain higher compression of the video signal in order to squeeze the video into the existing bandwidth.

III. UAV SIMULATOR DESCRIPTION

This chapter begins part two of this thesis and concerns itself with the UAV simulator. Part two consists of Chapters III, IV, and V. The objective in part two is to develop a low-fidelity UAV simulator designed to assist in the training of UAV operators. This chapter begins with a discussion of the design considerations that led to the implementation of the UAV simulator using two networked workstations. Additionally, it provides an overview of the UAV simulator and, lastly, this chapter discusses the problems encountered in trying to implement the UAV simulator on a single workstation. Chapters IV and V describe the UAV's two major components, the air vehicle operator and mission payload operator modules, in greater detail.

A. DESIGN CONSIDERATIONS

This section discusses the key factors considered in the design of the UAV simulator. These key factors being physical, operational, and implementation considerations. The physical considerations focused on the need to accurately duplicate the command and control mechanism of the UAV as well as the user interface. Operational considerations focused on the ability to make the simulator easily reconfigurable in terms of virtual world and threat scenarios. While implementation considerations focused on choosing the appropriate hardware/software configuration for creating three dimensional virtual environments in real time.

1. Physical Considerations

The two major physical considerations for developing the UAV simulator were the physical configuration of the simulator and defining the appropriate user interface. The physical configuration concerns itself with duplicating the division of labor used to control an actual UAV. The primary consideration for the design of the user interface was to replicate the functionality of the actual interface used in the Ground Control Station. This section discusses these physical considerations.

27

The physical separation of duties between the Air Vehicle and Mission Payload Operators determined the need to implement the UAV simulator using two workstations. As described in Chapter I, a deployed UAV is controlled via radio link by its Ground Control Station. Control of the deployed UAV is accomplished through the coordinated effort of two operators. The Air Vehicle Operator (AVO) is responsible for the operation of the UAV. The Mission Payload Operator (MPO) controls the sensor package on board the UAV. For the purpose of this thesis, the payload is assumed to be a black-and-white video camera but the UAV is also capable of carrying either an infra-red camera or a radio relay package. Because of this two-person command and control mechanism for the UAV, the architecture of the UAV simulator uses two workstations. The UAV simulator consists of one workstation for the AVO and the second workstation for the MPO. The two workstations communicate over an Ethernet LAN using the Distributed Interactive Simulation (DIS) protocol.

The design of the user interface involved attempting to duplicate the functionality of the actual user interface employed within the Ground Control Statio. As stated above, the AVO controls the flight of the UAV to and from the target area. The AVO controls the heading, pitch, and roll of the UAV using a joystick. The speed of the UAV is controlled by a knob. The MPO also uses a joystick to control the slewing of the UAV payload. Because both operators use a joystick, similar controls on the UAV simulator are implemented using the Thrustmaster Flight Control System Mark I.

2. Operational Considerations

In order to satisfy the need to have one simulator for many possible environments, the UAV simulator was developed so that the terrain model and threat environment would be reconfigurable. The simulator is designed to use Defense Mapping Agency (DMA) Digital Terrain Elevation Data (DTED) products in order to create the two-dimensional terrain display for the AVO module as well as the three-dimensional virtual world for the

28

MPO module. The concept being that provided with a digital DMA DTED product for a given piece of terrain, the simulator is capable of creating the appropriate two and three dimensional displays for the AVO and MPO modules, respectively. This thesis used the DMA DTED map data for Monterey, California but it is possible to load another map by simply changing the name and location of the DMA file at the command line when invoking the simulator.

Threat environments are also reconfigurable in that threat entities can be loaded into the world and the UAV simulator can test the operator's ability to perform in a threat environment. The changing of the threat environment requires programming and recompilation of the source code. Threat entities can be added to the virtual world and the simulator detects if the UAV operator flies within the threat entity's weapons range. This is implemented by using the collision detection mechanism provided in the Performer/SGI programming environment. For example, air defense threats can be placed in the synthetic environment. If the AVO flies the UAV inside of the air defense weapon system's threat envelope, then the threat entity fires on the UAV. Upon impact, the simulation begins anew and the AVO can perform another mission attempt based on lessons learned from the previous engagement.

3. Implementation Considerations

The purpose of this section is to explain why IRIS Performer from Silicon Graphics was used to implement the UAV simulator. First, this section describes IRIS Performer and its capabilities then discusses the implementation considerations which led to the decision to use Performer as the API.

IRIS Performer on a Silicon Graphic's Reality Engine Onyx Series computer provides the appropriate software/hardware mix to achieve the real time graphical displays necessary for the UAV simulator. Performer is an API which implements high performance graphics on Silicon Graphics products. It performs culling, multiprocessor

synchronization, intersection testing, and texturing in hardware [SGIF94]. The use of Performer in the UAV simulator allows for the rendering of the three dimensional terrain for the MPO module in real time. This is accomplished using Performer's culling feature where only those portions of the terrain visible from the UAV at a given point in time are drawn. Also, the use of Performer allows the UAV simulator to conduct collision detection in real time. This allows for real time intersection testing necessary to determine if the UAV has crashed into the ground or to determine if the UAV has been engaged by an air defense weapon system.

From the physical and operational considerations mentioned above, stringent requirements are levied upon a graphics applications program interface. The API must be capable of culling the terrain database and rendering the appropriate three-dimensional view as well as be capable of performing collision detection - all in real time. These are supported in IRIS Performer. For example, in order to create the terrain model, the API must create the corresponding geometry for 1201 x 1201 data points which are read in from the DMA DTED data file. These 1,442,401 elevations from the DTED data are used to construct a digital elevation map for the corresponding map sheet. Since computers are limited in the total number of polygons that can be drawn in any given scene, a decision must be made on which portion of the digital elevation map to draw. This is known as culling the database. Using the DTED map sheet as an example, it is not necessary to draw the entire map sheet in every frame. Rather, a more efficient implementation results from only drawing what the UAV can actually see from a given location. This culling function is executed by Performer allowing the UAV to traverse the virtual world in real time.

B.   SIMULATOR OVERVIEW

This section describes the basic architecture for the UAV simulator. As mentioned above, the simulator consists of two modules, the AVO and MPO.Figure 2 illustrates the architecture of the UAV simulator. The AVO module, located to the right in Figure 2,

displays a two dimensional map based on the map sheet of interest. This two-dimensional map sheet provides spatial orientation for the UAV pilot. In order for the operator to maneuver against a target, the AVO module allows the user to interactively change the speed, heading, and altitude of the UAV. The AVO maintains spatial orientation since the UAV's location is represented graphically as being at the head of the "snail trail". Figure 2 illustrates this point by indicating the UAV's location on the AVO display as being represented by the arrowhead at the end of the dotted line. Based on the UAV's location and attitude, the AVO module communicates with the MPO module using the DIS protocol over an Ethernet LAN. The UAV's position and orientation is encapsulated into a DIS entity-state protocol data unit (PDU), using library routines defined in [ZESW93] and is sent over the network. The MPO module, which is listening to the network, parses the PDU and provides the information to the MPO module. Using the UAV's position and orientation from the AVO module, the MPO module interprets this information as the eyepoint and renders a three-dimensional representation as depicted in Figure 2. This architecture clearly separates the AVO and MPO functions and duplicates the physical command and control mechanism for the UAV. A more detailed explanation of the AVO and MPO modules is provided in Chapters IV and V, respectively.

C. DESIGN ISSUES

This section discusses the design issues encountered during the development of this UAV simulator. The design issues involved the choice of the physical architecture, choice of input device, and the design of the user interface. The issue concerned with the choice of the physical architecture was whether to implement on one workstation or two. The issue with the choice of an input device was whether to use the keyboard or the Thrustmaster Flight Control System Mark I. Lastly, the issue with the user interface was whether to reproduce the actual user interface or to provide a user interface that was functionally equivalent.
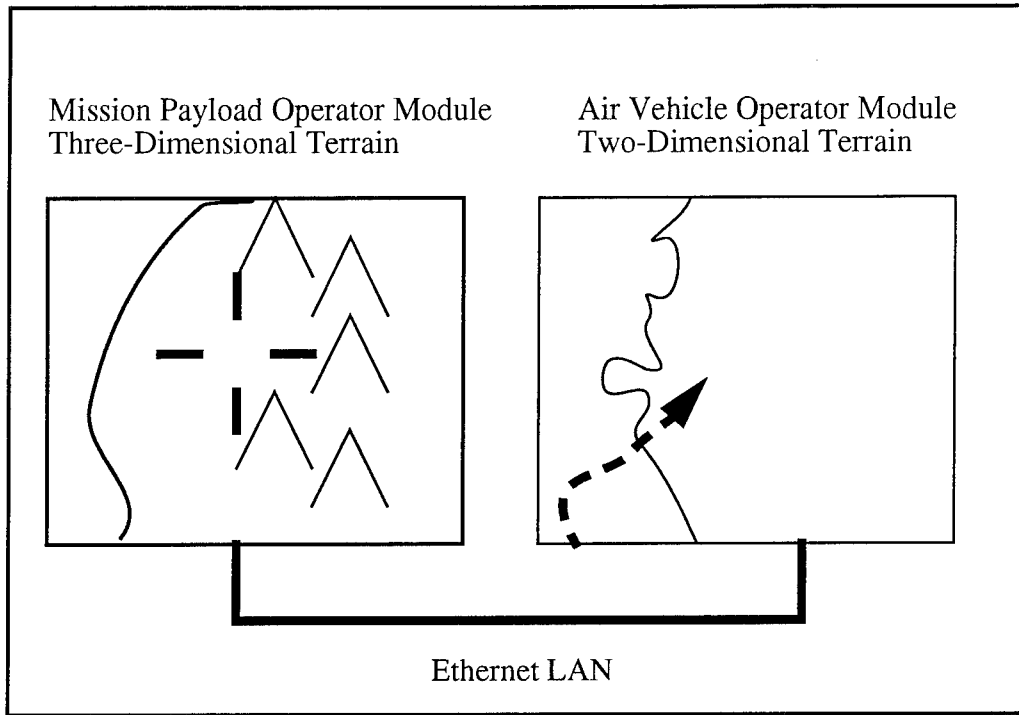
Figure 2: UAV Simulator Architecture

1. One Workstation Versus Two

The initial design of the UAV simulator called for the implementation of the UAV simulator on a single workstation. The goal was to implement what is now the AVO module in one window and implement the MPO module in another window with both modules sharing a common control panel also using Performer. The intent was to use the "video-out" option for the Reality Engine to create a synthetic video feed for testing the video distribution model. In other words, take the MPO module's display and send it out over the network as a means of testing a LAN architecture's ability to distribute digital

video. This would provide a means of testing a network's ability to process digital UAV video without having to fly an actual UAV. This architecture is depicted in Figure 3.
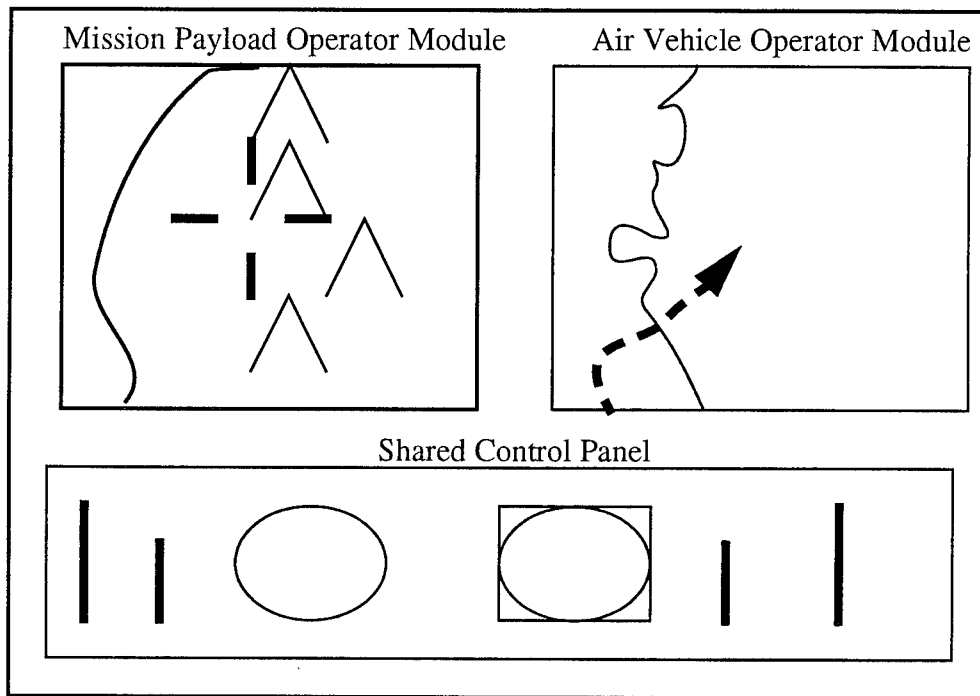
Figure 3: Single Workstation Simulator Design

The hardware used to implement this design had only a single graphics pipeline. The graphics pipeline is needed to render both the AVO and MPO displays. While it was not a problem to show either the MPO or AVO displays independently, the only way that both displays could be presented on the screen simultaneously was by putting Performer in single processor mode. This was accomplished by using the pfMultiprocess(APP_CULL_DRAW) command. The problem with this approach was that it took too long to render the scene and did not produce real time graphics. Since the simulator defined custom draw routines for both the AVO and MPO modules, Performer would attempt to draw both scenes at each iteration through the draw loop. This

unsuccessful attempt at implementing the UAV simulator on a single workstation using a single graphics pipeline was another consideration for implementing the simulator on two workstations.

2. Input Device And User Interface

The purpose of this section is to discuss design considerations with respect to input device and the user interface on the UAV simulator. The goal of any simulator should be to duplicate the environment of the user on the actual system. The UAV simulator developed in this thesis shares this goal but falls somewhat short. By using the flight control system, the input device on the AVO and MPO is duplicated but the control widgets used in the simulator versus what the users see in the Ground Control station are different. The simulator does however duplicate the basic functionality of the Ground Control station.

D. SUMMARY

This chapter discussed the rationale behind the design of the UAV simulator as well as the considerations which led to its implementation on two workstations using IRIS Performer. Also, this chapter provided an overview of the UAV simulator and describes how the AVO and MPO modules interact using the network. Additional information on the actual working of the UAV simulator is provided at Appendix A. The AVO and MPO modules are discussed in greater detail in the next two chapters. Chapters IV and V provide an overview of the functionality of the two modules then discusses in detail how they work.

IV. AIR VEHICLE OPERATOR MODULE

This chapter describes the major components of the AVO module. We begin with a description of the AVO graphical display which is followed by a discussion of the major components of the module. More specifically, the chapter describes how the AVO module generates the two-dimensional terrain display, its use of overlay planes for creating the AVO module's graphics, the flight dynamics model, and how the AVO module uses the DIS protocol for communicating with the MPO module.

A. AVO GRAPHICS DESCRIBED

The AVO module controls the flight of the UAV within the synthetic environment. Figure 4 depicts an actual display from the AVO captured by doing a "screen-grab" during a UAV simulation.This simulation was performed using the DMA data for Monterey, California. The red line which begins in the lower left-hand corner depicts the flight path of the UAV during this simulation. The red line terminates on the Monterey Peninsula and denotes the current location of the UAV during this mission. The UAV's attitude is packetized into an entity state protocol data unit and is broadcast over the Ethernet LAN.

The UAV is controlled using the Thrustmaster Flight Control System Mark I. The AVO controls the speed of the UAV using a throttle and controls the attitude of the UAV by using a joystick. The AVO changes the speed and direction of flight of the UAV interactively.

B. GENERATING THE TWO-DIMENSIONAL TERRAIN MODEL

The AVO module terrain model is generated directly from DMA DTED data. Given a digital DMA map file, the AVO module reads in the elevations and creates a Digital Elevation Map (DEM). From the DEM, the AVO module reads the data points and creates the two-dimensional terrain model coloring points based on elevation. Drawing a triangle requires three vertices. Since the terrain is generated as a large number of triangles, the amount of data required to draw a series of triangles can be reduced by using a drawing
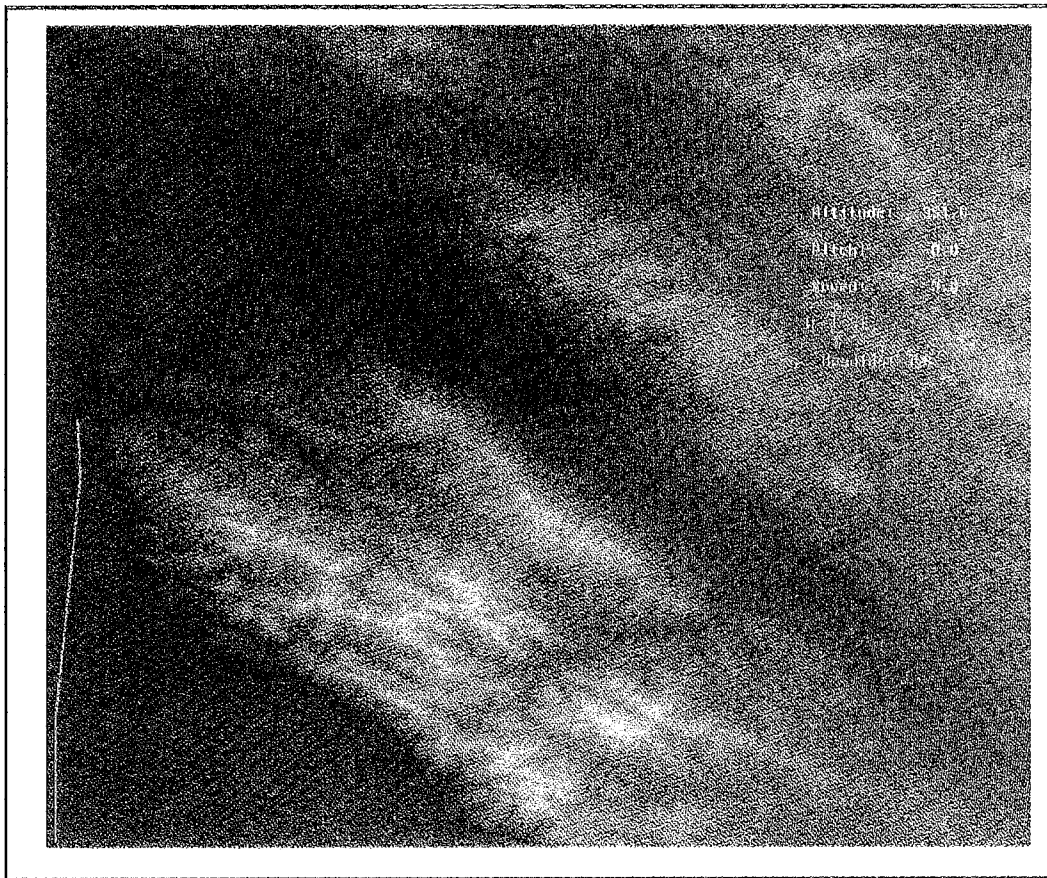
Figure 4: The Air Vehicle Operator Display

primitive known as a triangle mesh. The triangle mesh reduces the amount of data required for drawing a large number of triangles by arranging the vertices in such a manner that the drawing of a triangle "n+1" uses two vertices used to draw triangle "n". Once a particular column has started, a triangle mesh uses two vertices from the previously drawn triangle. This effectively reduces the amount of information needed to draw a series of triangles. Figure 5 illustrates this concept. Triangle "A" consists of vertices 0, 1, and 2. By using a triangle mesh, only vertices 3, 4, and 5 are needed to draw the following three triangles "B", "C", and "D". The triangle mesh drawing primitive permits the efficient drawing of the geometry for the AVO module. The AVO module's two-dimensional terrain representation is drawn using the NORMALDRAW plane. The additional information in the AVO

graphical display is created by "overlaying" the other information on top of the terrain representation. The following section discusses this use of graphical overlay planes.
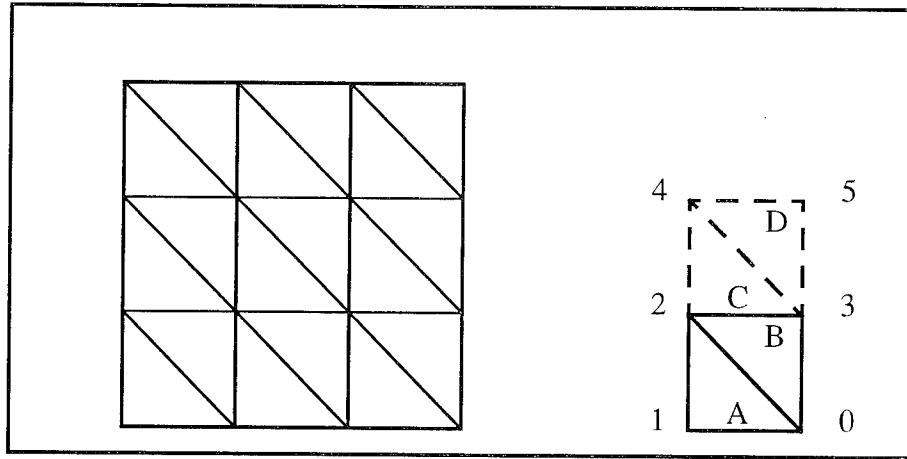
Figure 5: Generating A Triangle Mesh

C. GRAPHICAL OVERLAY PLANES

In addition to the shaded map, the AVO graphical display includes a "snail trail" used to indicate the present location of the UAV as well as provides a Heads Up Display (HUD) to keep the AVO informed of key parameters. This section describes first in general terms the use of overlay planes then describes their implementation in the AVO module.

Overlay planes are used to "layer" information onto the graphical display. In general terms there are three basic overlay planes available to the programmer. These are the underdraw, normaldraw, and overdraw planes. Associated with each of the overlay planes is a frontbuffer and backbuffer. This provides a total of six planes on which to draw. The AVO module uses these overlay planes to produce the AVO graphical display. At the lowest level, the terrain is drawn using the backbuffer of the "normaldraw" plane. On top of the terrain information, the snail trail is drawn using the frontbuffer of the normaldraw

plane. The heads up display is drawn on top of the normaldraw plane using both the front and back buffers of the overdraw plane.

The AVO picture provides the operator with such information as the speed, altitude, pitch and direction. All of these are displayed textually on the AVO screen. The AVO HUD also includes a compass rose to provide the AVO operator with heading information graphically. These are drawn and updated using the overdraw plane. Because this information is updated dynamically as the position of the UAV changes, it requires the swapping of both buffers in order to prevent a strobing effect with the textual information.

D. THE FLIGHT DYNAMICS MODEL

The AVO module allows the user to control the flight of the UAV interactively. The operator changes the heading, pitch, and roll of the UAV using either the keyboard or flight control sticks. For example, Table 2 lists the keyboard keys and their effect on the flight path of the UAV.

Keypad Key	Result of Keypad Key Input
+	Increase UAV speed.
-	Decrease UAV speed.
8	Increase altitude of UAV.
2	Decrease altitude of UAV.
6	Change UAV heading to the right.
4	Change UAV heading to the left.
9	Increase roll to the right.
3	Increase roll to the left.
ESCAPE	Terminate simulation and exit program.

TABLE 2: AVO Module Keyboard and Function Mappings

The UAV Simulator is not physically based and uses a simplistic flight model. The reason for this is twofold. First, there are several variations of UAVs. Once a model is chosen, the appropriate flight dynamics can be integrated into the simulator. Secondly, the flight models for the Hunter and Pioneer UAVs are still in the development phase.

E. COMMUNICATING POSITIONAL DATA OVER THE NETWORK

In its current configuration, the AVO module communicates with the MPO module ten times per second using the Distributed Interactive Simulation protocol. In order to packetize the UAV positional data, the AVO module puts its position and orientation data into the DIS entity-state protocol data unit. This information is packetized using the sendentitystate protocol data unit call defined in [ZESW93].

F. SUMMARY

This chapter discussed the Air Vehicle Operator module of the UAV simulator. The key functions of this module are to generate the terrain representation, provide the user with some key information, and to communicate with the MPO module. The key features of the MPO module are discussed in the next chapter.

V. THE MISSION PAYLOAD OPERATOR MODULE

This chapter describes the Mission Payload Operator (MPO) module. The primary features of this module are: to simulate the performance of the UAV's payload; to provide the user with information concerning the status of the UAV as well as it's payload; and to determine if the UAV has crashed or has been engaged by an enemy weapon system. We begin with an overview of the MPO module then discuss its major components. These major components are generate the three-dimensional terrain model, create the control panel, and determine collision detection.

A. MPO OVERVIEW

While the Air Vehicle Operator (AVO) module generates a location for the UAV providing spatial orientation to the operator by using a two-dimensional terrain model, the MPO module produces a three-dimensional view from the location generated by the AVO module. Therefore, the MPO module simulates what is seen by the UAV's payload. To illustrate this point, Figure 6 depicts the MPO display during the same mission as the AVO display shown in Chapter IV, Figure 4. The view in Figure 4 informs the AVO on the location of the UAV at that instant in time. The corresponding view in Figure 6 represents what the UAV's camera detected at that same instant in time.

In addition to rendering the view of the payload, the MPO module also provides the user with information concerning the status of the UAV and its payload by using a control panel. The control panel informs the MPO on critical UAV and payload parameters. This control panel provides the MPO with information concerning the speed, direction, and altitude of the UAV. Additionally, the control panel informs the user on the orientation, pitch, and focal length of the UAV's video camera.

One last feature of the MPO is its ability to perform collision detection. There are two general categories of hazards in the UAV's synthetic environment - natural and man-made. The terrain model provides a natural obstacle to the UAV since the aircraft could crash into the terrain. Enemy air defense assets constitute the man-made hazards. The AVO could

41

maneuver the UAV within the weapon's range of an enemy air defense asset which could engage the UAV. In order to properly train operators against these hazards, the UAV simulator must be capable of detecting when it has hit a stationary object, such as terrain, as well as determine if the UAV has been engaged by an enemy weapon system. Collision detection is implemented in the MPO module using IRIS Performer. Once the MPO module has determined that a collision has occurred, the MPO module sends a message to the AVO module to indicate that the UAV has been damaged and the simulation begins anew.
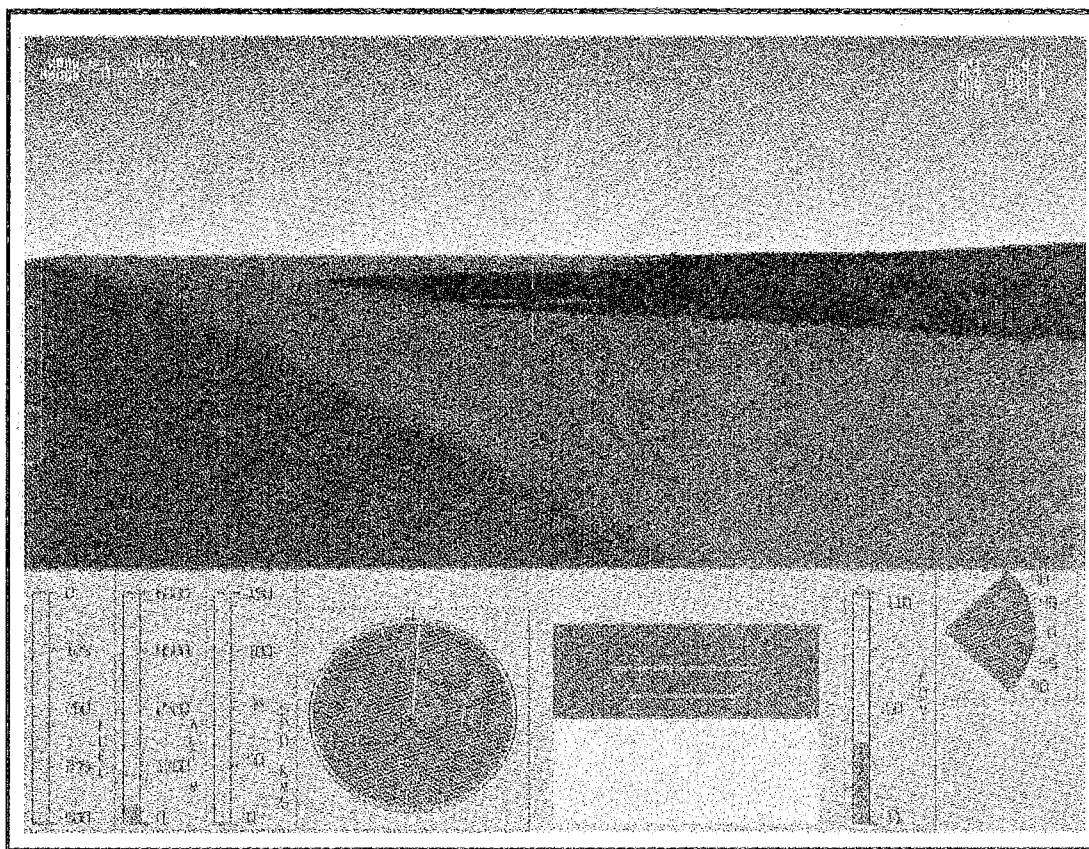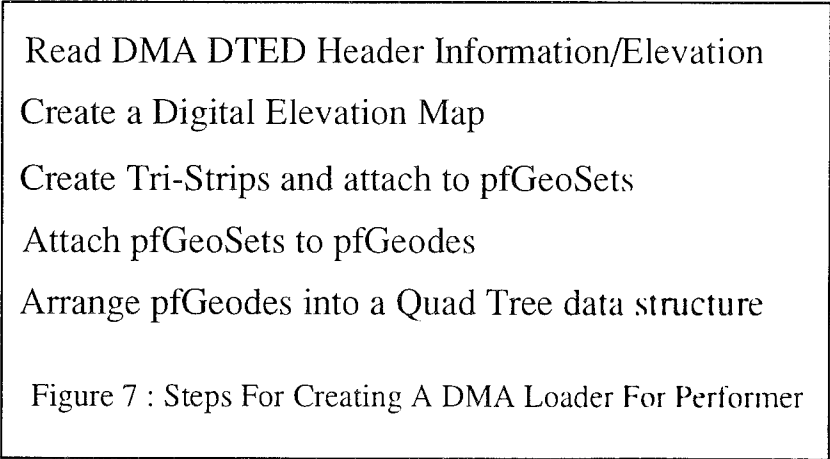
Figure 6 : The Mission Payload Operator Display

B. GENERATING THE THREE-DIMENSIONAL TERRAIN

In order for Performer to generate the three-dimensional terrain model directly from the DMA DTED data, this thesis developed a custom loader. The MPO module is designed to create the terrain model directly from any standard DMA DTED data. This loader is defined by the procedure *LoadDTED()* in the program *Load_DMA.c*. The loader essentially reads the DMA DTED data directly into IRIS Performer *pfGeodes* which are organized into a quad-tree data structure. This process enables Performer to read, cull, and display the DMA DTED data directly. The steps taken to create the loader are listed below in Figure 7.

Read DMA DTED Header Information/Elevation

Create a Digital Elevation Map

Create Tri-Strips and attach to pfGeoSets

Attach pfGeoSets to pfGeodes

Arrange pfGeodes into a Quad Tree data structure

Figure 7 : Steps For Creating A DMA Loader For Performer

The first step in the process of creating the loader is to read the DMA DTED header information. From the DMA DTED header, we obtain the maximum number of rows and columns in the data file, the maximum and minimum elevation, and the spacing between the data points. The maximum number of rows and columns is needed for generating the dimensions of the Digital Elevation Matrix (DEM). The DEM is a two-dimensional array that contains an elevation for each lattitude and longitude pair. The maximum and minimum elevations are used to create the color ramp for the terrain data. In this simulation, the color ramp is established by interpolating shades of brown between the maximum and minimum elevations. The higher elevations are represented by lighter shades of brown while the lower elevations are represented by darker shades of brown. This coloring

43

scheme holds true with the exception that elevations of zero are colored blue to represent water. The distance between data points is used to place the data points the proper distance apart based on the window size of the terrain representation.

The next step in the process of generating this loader is to create the quad-tree data structure to hold the geometry used to produce the three-dimensional terrain model. The first step in creating the quad-tree is to determine the basic building block for the data structure. The data file is successively subdivided until the basic element is a 75 x 75 point block of elevations. A DMA DTED file is represented as a 1201 x 1201 matrix of elevations [DEFE86]. At each successive level, the number of data points are successively halved beginning with 1201 data points along each axis and ending at 75 data points along each axis. This subdivision is illustrated in Figure 8. Because of the need to prevent seams between adjoining sections of the terrain, we must duplicate the data points along the seams. This makes the basic building block for the quad-tree a block of 76 x 76 elevation points.

Each 76 x 76 building block is then read, a column at a time, to create a tri-strip as the graphic primitive. Each of these 76 point tri-strips are attached to a *pfGeoSet*. Each *pfGeoSet* consists of 76 tri-strips and constitutes all of the leaf nodes for the quad-tree structure. Four adjoining *pfGeoSets* are attached to one *pfGeode* where *pf Geodes* become the internal nodes of the quad tree data structure. Consequently, the quad-tree is constructed from the bottom up with 256 leaf nodes, or *pfGeoSets*, and 85 *pfGeodes* or internal nodes.

C.   CONTROL PANEL

Depicted in Figure 6 is a set of instruments designed to inform the MPO on details concerning UAV and payload parameters during the execution of a mission. From left to right these instruments consist of a fuel gauge, an altimeter, a speed indicator, a heading indicator, an artificial horizon, a field of view indicator, and a pitch indicator. The fuel gauge, the altimeter and speed indicator provide the MPO with information on key UAV
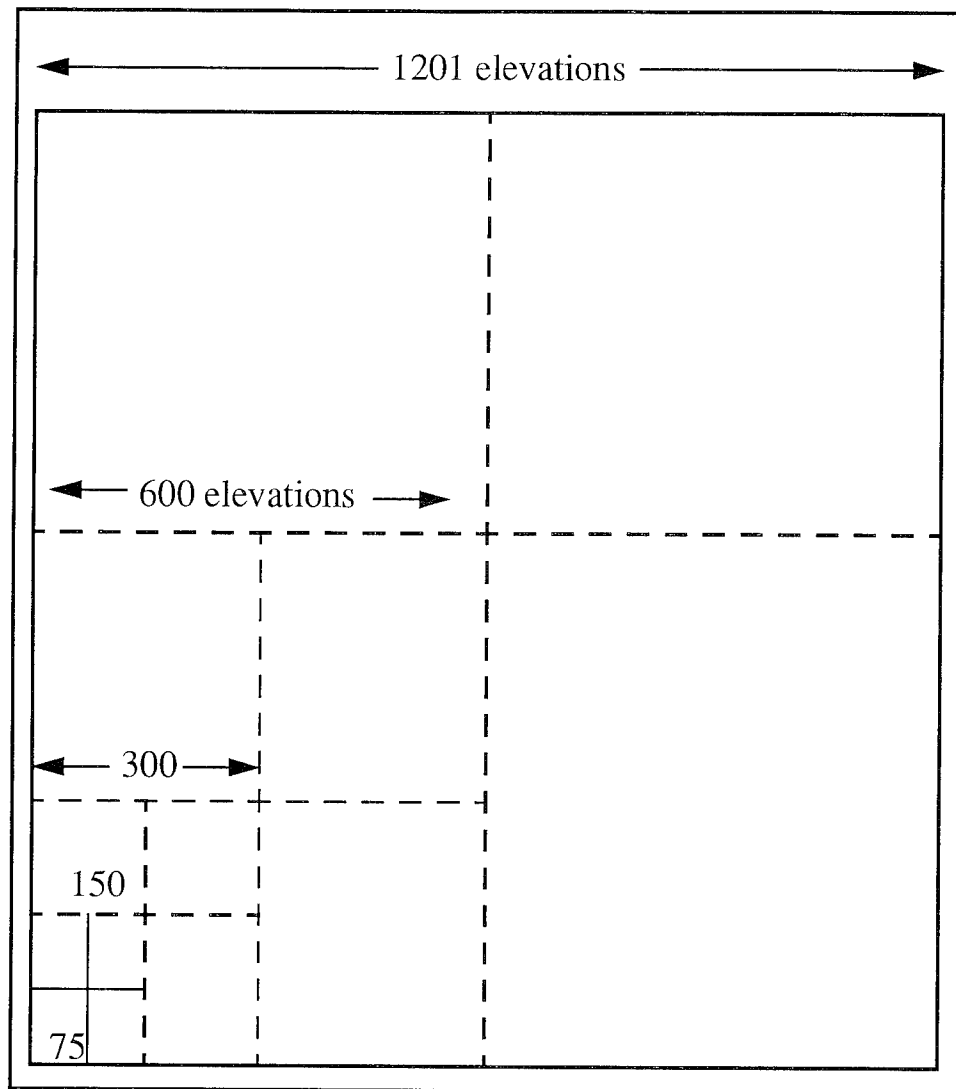
Figure 8: Subdividing The Terrain Data File

parameters and are self explanatory. The purpose of the compass is twofold. First, it provides the MPO with information on the heading of the UAV. Secondly, the compass provides the MPO with information on the orientation of the payload with respect to the aircraft. This is indicated by the red arrow. When the red arrow is aligned with the black

"tic" mark at the compass' twelve o'clock position, the heading of the UAV and its payload coincide. The next instrument, the artificial horizon, provides the MPO with information on the pitch and roll of the UAV. The next gauge shows the field of view of the MPO's payload and simulates the operation of the camera's zoom lens. Lastly in the upper right hand corner is a gauge that indicates the camera's angle of depression. Based on the camera's angle of depression this simulator calculates the range to target. The target is represented as the object in the "cross hairs" on the heads up display. The range to target information is displayed textually on the HUD's upper left hand corner beneath the UAV's lattitude and longitude. Since we know the altitude of the UAV and the depression angle of the camera, we can calculate the range to the object appearing in the "cross hairs" of the heads up display. This range to target is calculated using the formula derived in Figure 9.
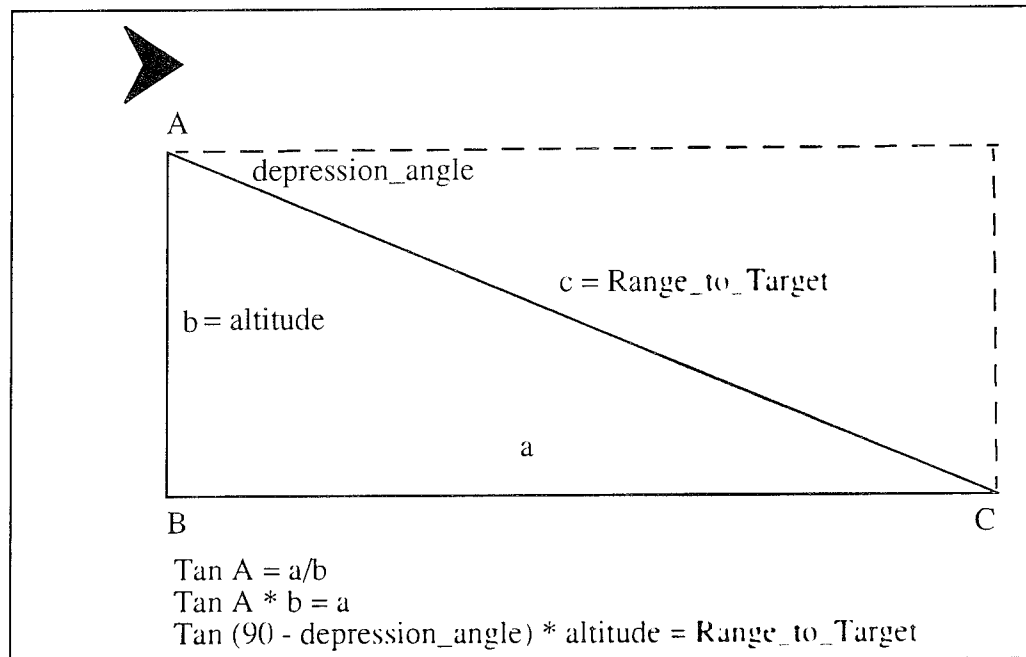
Figure 9: Deriving The Range-To-Target Formula

46

The control panel is one component of the MPO module that is not implemented using IRIS Performer. In fact, the MPO module saves state, exits IRIS Performer and draws the control panel using graphics subroutines. Upon completion of drawing the control panel with its instruments and needles, the module restores state to IRIS Performer and resumes. The code fragment depicted in Figure 10 is taken from the MPO module, program *mpo.c*, and illustrates leaving and restoring state to Performer. Lines one through nine are concerned with exiting Performer. The instruments are drawn by the procedures called in lines ten and eleven and lines twelve through seventeen illustrate the return path to Performer state.

```
 1.  pfPushState();
 2.  pfBasicState();
 3.  pfCullFace(PFCF_OFF);          Save and Exit Performer
 4.  zbuffer(FALSE);
 5.  pfPushIdentMatrix();

 6.  ortho2(-0.5, 1279.5, -0.5, 341.5);
 7.  zbuffer(FALSE);
 8.  cpack(C_INST_BK);
 9.  clear();                        Draw Control Panel
10.  draw_inst_back();
11.  draw_inst_ptr();

12.  pfPopMatrix();
13.  mmode(MPROJECTION);
14.  loadmatrix(ProjMat);            Restore Performer State
15.  mmode(MVIEWING);
16.  zbuffer(TRUE);
17.  pfPopState():
```

Figure 10 : Exiting and Returning To Performer For Drawing The Control

The instruments are drawn in two parts using a layered approach. First, the control panel, its instruments, titles and tic marks, are drawn using the procedure *draw_inst_back()*

47

defined in the program *draw_inst_bk.c*. Next, the needles and pointers of the control panel are layered on top of the control panel using the procedure *draw_inst_ptr()* defined in the program *draw_inst_ptr.c*.

D. COLLISION DETECTION

In this simulator, collision detection is used to determine whether the UAV has crashed into the ground or if the UAV has been engaged by an enemy weapon system. The simulator's use of IRIS Performer enables the detection of collisions in real time. This is accomplished by using Performer's intersection library routines as defined in [HART94]. Two intersection masks are used in this simulator, one to determine the UAV's collision with the ground and the second to determine whether or not the UAV has been shot down. IRIS Performer will execute intersection detection provided the user defines an intersection mask for an entity as well as establishes the line segments from the entity that needs to be tested. This simulator defines an intersection mask for the terrain as well as a line segment from the bottom of the UAV. If that line segment intersects with the underlying terrain primitives, Performer routines will inform the calling program that a collision has occurred. In this UAV simulator, once a collision is detected, the MPO module communicates to the AVO module that the UAV has crashed. This collision message returns both the AVO and MPO modules to the start point of the simulation.

This section briefly describes how collision detection is achieved in this simulation using Performer. IRIS Performer defines library routines for establishing intersection masks. The *pfNodeTravMask* routine tells Peformer to establish a terrain mask for the polygons which define the terrain in this UAV simulator. Later, the simulator defines a line segment that extends from beneath the UAV. The MPO module queries the *pfSegIsectNode* routine to determine if the line segment has intersected with the terrain intersection mask. If so, then the MPO module notifies the AVO module to start over using the DIS entity-

48

state protocol data unit. A similar mask, line segment relationship is defined between the ADA asset and the UAV for determining whether or not the UAV has been shot down.

VI. CONCLUSIONS AND TOPICS FOR FUTURE RESEARCH

A. CONCLUSION

The purpose of this thesis was to recommend two enhancements to the Unmanned Aerial Vehicle System. The first enhancement recommends an architecture that will allow the distribution of full-motion video across a local area network. The second enhancement develops a UAV simulator for training new operators and developing doctrine for this relatively new weapon system.

At the beginning of this investigation, there was much speculation as to whether or not it was possible to network full-motion video. It was assumed that full-motion video would be necessary in order to fully satisfy maneuver commander requirements. After researching the present system for video distribution, this thesis concludes that distributing digital video across a local area network is feasible and recommends an architecture that will distribute full-motion UAV video throughout the tactical operations center.

Lastly, this thesis analyzed the UAV Ground Control System and produced a UAV simulator. It produced a UAV simulator which emulates the command and control of this system by using two, networked workstations to replicate the functions of the air vehicle and mission payload operators. The simulator can establish its terrain model directly from standard DMA DTED products. Potential uses for this simulator include: the training of new operators; as a tool for the development of UAV employment concepts; as a tool to conduct pre-mission planning or post-mission analysis for UAV training.

The following sections outline the conclusions drawn from this thesis:

1. Video Distribution Model Conclusions

Distributing live UAV video across a local area network within the tactical operations center is absolutely possible using commercially available products. This capability is being developed in the commercial sector as part of the digitization of the

television industry and video-on-demand initiatives. This rapidly emerging technology has many potential military applications. One of which is to support digital UAV video. When considering these systems for possible military use, the center of gravity is the choice of a compression technique. The choice of a compression technique determines the source interchange format, network topology, storage requirements, and playback capabilities.

This thesis recommends an architecture that is optimized for intelligence purposes. It provides for: full NTSC SIF, frame-by-frame playback, and multicast, full-motion video across the local area network. Each of these are the most demanding requirements possible for each category. Because it is the most demanding, it is also the most expensive. Trade-offs for less expensive alternatives can be made at the expense of reducing the amount of information available from UAV video.

While distribution throughout the TOC is within the realm of the possible, the technology for sharing full-motion UAV video with subordinate and higher commands is not a present capability. This limitation results from the reduced bandwidth available using the communications systems currently fielded to United States Army units.

2. UAV Simulator Conclusions

A simulator can be a valuable asset for: training new operators, teaching commanders how to use UAVs, and for developing UAV doctrine. There is currently no simulator available for the training of UAV operators. A simulator could provide more flight time for UAV crews without the corresponding costs and wear and tear on actual UAVs. Additionally, the simulator could be used to train operators in UAV employment and search techniques. Also, the UAV simulator could be used to train commanders on how to integrate UAVs into their maneuver plan. For example, prior to a rotation to the National Training Center, a commander could use the UAV simulator as a terrain visualization tool for developing his scheme of maneuver. Lastly, another possible use of the simulator would

be to develop doctrine for UAVs. This is a new system to US Army units and a simulator could be used to test the applicability of UAVs in various scenarios.

B. TOPICS FOR FUTURE RESEARCH

Several topics for further study can be derived from this thesis. They can be categorized by their relevance to either the video distribution problem or the UAV simulator.

This thesis recommends an architecture for distributing live video. Will this architecture perform as claimed? Can ATM support multiple motion JPEG video streams simultaneously?

Is motion JPEG compression actually more desirable to a maneuver commander than MPEG? Is the reduced cost to implement the video distribution model using MPEG worth the sacrifice in terms of SIF and frame-by-frame playback capability?

Assuming that a real-time wavelet or fractal compression coder decoder is available, how can these compression techniques be applied to the video distribution model?

The UAV simulator developed by this thesis is not physically based. The simulator could be transformed into a high fidelity simulator with the incorporation of a flight dynamics model for the Hunter model UAV into the simulator.

The integration of the UAV simulator into NPSNET would provide the first virtual intelligence collection platform in NPSNET. Based on returns from the UAV, entities could be placed on the HUD instead of the current method of providing a omniscient view of battlefield entities to all players. The integration of intelligence collection and dissemination is needed for high-fidelity combat simulations.

53

UAV SIMULATOR

USER'S GUIDE

Advisor: Dr. David R. Pratt

Written by: Frank Tipton

Table of Contents

UAV Simulator Description

This section describes the basic architecture and key features of the Unmanned Aerial Vehicle (UAV) simulator. The simulator consists of two components, the Air Vehicle Operator (AVO) and Mission Payload Operator (MPO) modules. The UAV simulator is an interactive, networked simulator that uses the Distributed Interactive Simulation (DIS) protocol for communicating between workstations. It is terrain independent in that provided with a standard Defense Mapping Agency (DMA) Digital Terrain Elevation Data (DTED) product, the UAV Simulator can produce a two-dimensional and three-dimensional representation of a DMA DTED data file. It implements collision detection which allows for the simulator to determine if the UAV has crashed into the ground or if the UAV has been engaged by an enemy Air Defense Artillery (ADA) weapon system. This simulator requires two, networked Silicon Graphics workstations running the IRIS Performer Applications Programming Interface, version 1.2, on the IRIX operating system, version 5.2. The input devices for both modules can be either the *Thrustmaster Flight Control System Mark I* or the workstation's keyboard.

The AVO module provides the user with a two-dimensional representation of the target area as well as provides the user with feedback on key UAV parameters such as speed, altitude, and heading. It provides spatial orientation to the UAV pilot using a two dimensional map based on the area of interest. The AVO module allows the user to interactively change the speed, heading, and altitude of the UAV. This can be done using either Flight Control System Mark I's from Thrustmaster or by using the workstation's keyboard. The UAV's location is represented graphically as being at the head of a "snail trail". At the start of each mission, the simulation begins with the UAV's position as being in the lower left hand corner of the graphics window. As the AVO changes the speed, heading, and pitch of the UAV, the "snail trail" illustrates the movement history of the aircraft. Based on the UAV's location and attitude, the AVO module communicates with the MPO module

using the DIS protocol over an Ethernet Local Area Network (LAN). The UAV's position and orientation is encapsulated into a DIS entity-state protocol data unit (PDU) and is sent over the network. The MPO module, which is listening to the network, parses the PDU and provides the information to the MPO module's main simulation loop.

Using the UAV's position and orientation from the AVO module, the MPO module interprets this information as the eyepoint and renders a three-dimensional representation of what the UAV's payload can see from that location. The MPO also provides a control panel which informs the MPO of key UAV and payload parameters. The MPO module allows the operator to slew the UAV's camera, manipulate the camera's pitch, as well as to increase and decrease the camera's field of view. As in the AVO module, the controls can be either the workstations's keyboard or the Flight Control System Mark I from Thrust-master. Additionally, the MPO module implements collision detection. It can determine if the UAV has flown into the ground or if it has been detected by an enemy ADA weapon system. Once the MPO module determines that a collision has occurred, it sends a message to the AVO module that the UAV has crashed and the AVO module will start the simulation over again. It keeps the previous "snail trail" on the AVO display and begins a new "snail trail" in another color.

Directory Structure and Program Files

The UAV Simulator is broken down into the following directories and files. Figure A-1 shows the directory structure of the UAV simulator. As stated above, it consists primarily of the AVO and MPO modules. Figure A-2 and Figure A-3 show the corresponding files within the AVO and MPO directories, respectively.

```
┌─────────────────────────────────────────────────┐
│                                                 │
│                  ┌──────────┐                   │
│                  │   UAV    │                   │
│                  └──────────┘                   │
│              ┌──────┘    └──────┐               │
│        ┌──────────┐        ┌──────────┐         │
│        │   AVO    │        │   MPO    │         │
│        └──────────┘        └──────────┘         │
│                                                 │
└─────────────────────────────────────────────────┘
```
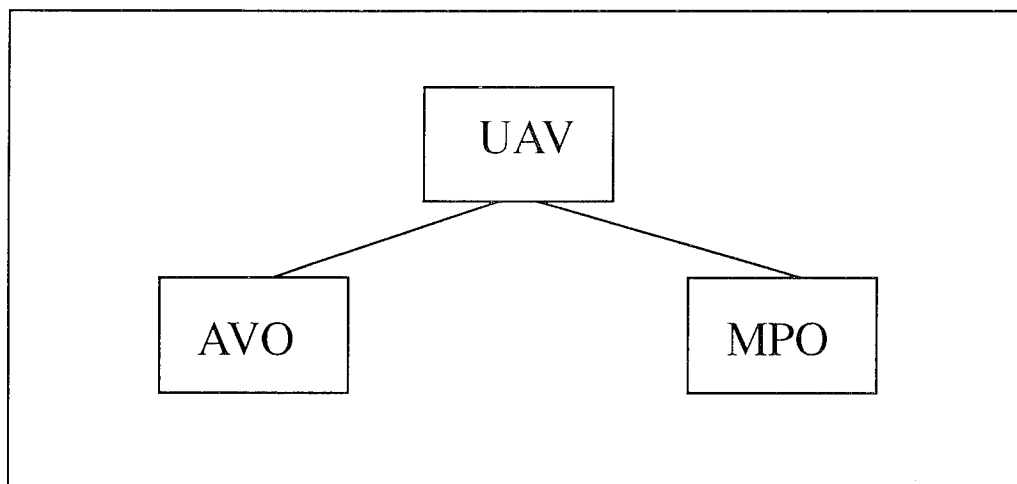
Figure A-1: The UAV Simulator Directory Structure

Air Vehicle Operator Module

The next diagram, Figure A-2, illustrates the organization of the programs that constitute the Air Vehicle Operator (AVO) module.
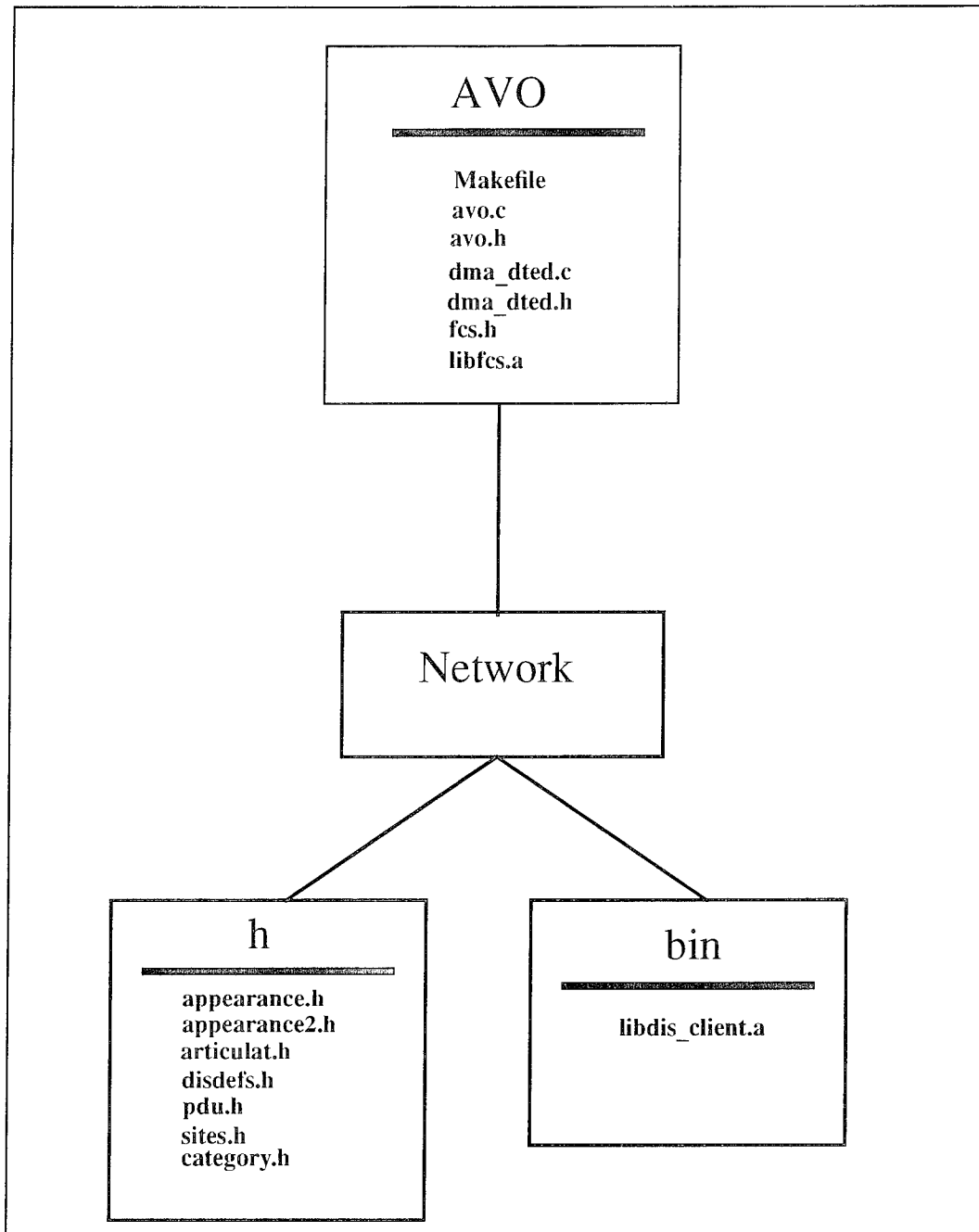
Figure A-2: The AVO Module Directory Structure

Contents of the AVO Directory

avo.h   Header file for *avo.c*

avo.c   Program for AVO module. Creates 2D map, sends and receives DIS PDUs, performs collision detection, and draws heads up display.

dma_dted.h   Header file for *dma_dted* program.

dma_dted.c   Program for reading DMA DTED terrain data.

fcs.h   Header file for use of **THRUSTMASTER Weapons Control System Mark I.**

lib_fcs.a   Library file containing object code for use of **THRUSTMASTER Weapons Control System Mark I.**

Contents of the AVO/Network/h Directory

*.h   All of these are header files that are used to implement the various PDU types for DIS.

Contents of the AVO/Network/bin Directory

libdis_client.a   Library file for implementing the DIS networking library routines.

Contents of the Mission Payload Operator Module

Load_DMA.c   This program reads the DMA DTED file and loads the information into IRIS Performer geometry primitives. This allows Performer to cull the database in real time.

dma_dted.c   This program provides the functions that allows *Load_DMA.c* to read DMA DTED data.

dma_dted.h   Header file for *dma_dted.c*.

fcs.h   Header file to allow *mpo.c* to use **THRUSTMASTER Weapons Control System Mark I.**

hosts.dat   This data file contains a list of the servers that the UAV Simulator can connect to.

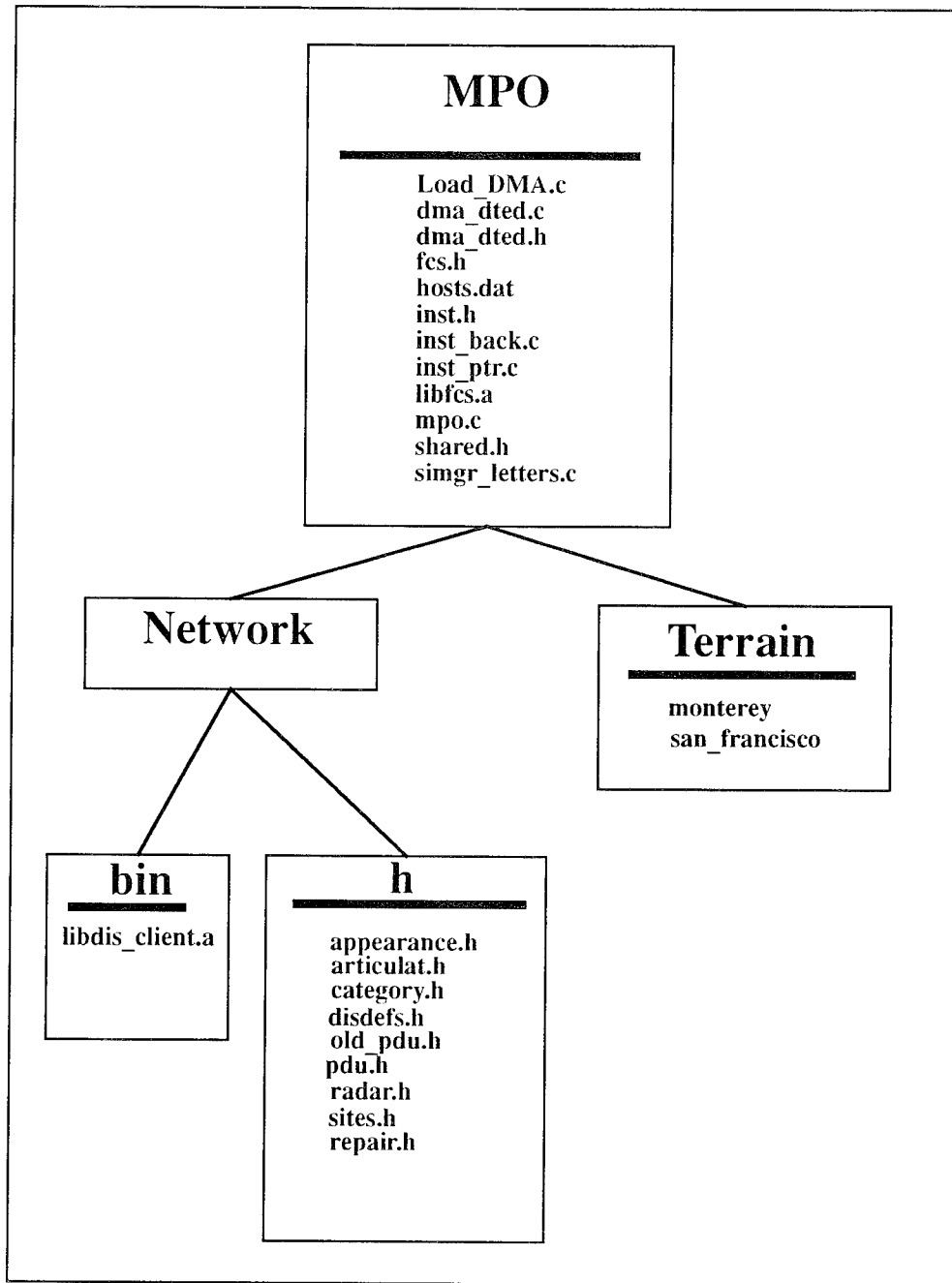**inst.h**   Header file for control panel variable declarations.

```
                    ┌─────────────────────────┐
                    │          MPO            │
                    │      ─────────────      │
                    │                         │
                    │      Load_DMA.c         │
                    │      dma_dted.c         │
                    │      dma_dted.h         │
                    │      fcs.h              │
                    │      hosts.dat          │
                    │      inst.h             │
                    │      inst_back.c        │
                    │      inst_ptr.c         │
                    │      libfcs.a           │
                    │      mpo.c              │
                    │      shared.h           │
                    │      simgr_letters.c    │
                    └─────────────────────────┘
                       /                    \
        ┌─────────────────────┐      ┌─────────────────────────┐
        │      Network        │      │        Terrain          │
        └─────────────────────┘      │     ──────────────      │
              /        \             │       monterey          │
                                     │       san_francisco     │
  ┌─────────────┐   ┌──────────────┐ └─────────────────────────┘
  │     bin     │   │      h       │
  │  ─────────  │   │  ─────────   │
  │ libdis_client.a │ appearance.h │
  │             │   │ articulat.h  │
  └─────────────┘   │ category.h   │
                    │ disdefs.h    │
                    │ old_pdu.h    │
                    │ pdu.h        │
                    │ radar.h      │
                    │ sites.h      │
                    │ repair.h     │
                    └──────────────┘
```

Figure A-3: MPO Directory And File Structure

inst_back.c   This program draws the back of the instrument control panel to include panel widgets and lettering.

inst_ptr.c   This program draws the pointers to the widgets drawn on the control panel.

lib_fcs.a   This library routine enables the use of the use of the **THRUSTMASTER Weapons Control System Mark I.**

mpo.c   This program is the driver program for the MPO module. It listens to the network for an update from the AVO module, draws the 3D terrain model, allows the user to interactively control the UAV payload and determines if the UAV has crashed or has been shot down.

shared.h   This is the header file used for all shared variables in the MPO module.

simgr_letters.c   This program is referenced by *inst_back.c* for putting text and letters on the control panel.

Installation

To obtain the programs for this simulator, download the simulator's source code over the Internet using the file transfer protocol (FTP). In order to acquire the needed data files, create a directory for this simulator, and change to that directory. This sequence is performed by using the following commands:

```
mkdir UAV
cd UAV
ftp elsie
cd ~tipton/Public
binary
get UAV.tar.Z
quit
```

Now un-compress and un-TAR the program files *UAV.tar.Z* and *UAV.tar* then do a listing of the directory with the following commands.

```
uncompress UAV.tar.Z
tar -xovf UAV.tar
ls
```

Now you should be ready to run the simulator provided that you have the proper hardware and software configuration. The MPO module requires a Silicon Graphics Reality Engine II using Performer version 1.2 and the IRIX operating system, version 5.2. These workstations must be connected with an ethernet network. The *hosts.dat* files in the MPO and AVO directories must be modified in order for the networking code to work. Each entry in the *hosts.dat* file contains the host name, site number, machine number, and the ethernet interface number that the machine uses to communicate over the network.The *host.dat* must be modified to reflect the configuration of the network at the new site. As for the input device required for the UAV Simulator, flight control sticks are optional. The

65

simulators will run, as is, using the workstation's keyboards.

Now that the files have been transferred, you have verified that you are configured correctly, and have modified the hosts.dat file, you should be ready to begin. The next section will describe how to use the UAV Simulator.

How To Use The UAV Simulator

The UAV Simulator is designed to train the mission payload and air vehicle operators to work together as a team against assigned targets. The AVO module allows the AVO to fly the UAV against a target while the MPO module allows the payload operator to control the video camera on-board the UAV. Together, the two operators fly the UAV against assigned targets. In this section, we discuss using the UAV Simulator on a reconnaissance mission using a synthetic environment. Both the AVO and MPO must run the same terrain database for the simulation. This example uses the San Francisco terrain database with an enemy ZSU 23/24 weapon system located on Alcatraz Island which is located in the San Francisco bay. This UAV Simulator comes with two terrain databases. One terrain database is of the San Francisco, California area and the other is of Monterey, California.

To begin the simulation, both operators must be located in their respective directory. The AVO must be in the ~UAV/AVO directory and the MPO must be in the ~UAV/MPO directory. With both operators in these directories, then they can type *'START_MPO_SAN_FRANCISCO'* or *'START_AVO_SAN_FRANCISCO'* at the prompt. These are text files that show the command that can be either typed in at the next prompt or can be highlighted and entered using the workstation's mouse. This will start all preprocessing for its respective module. The MPO operator may have to also initialize the flight control sticks. The flight control stick initialization sequence is program driven and the user needs only to move the controls and press enter as prompted by the program. Once the AVO module has started, the AVO display will resemble Figure A-4.
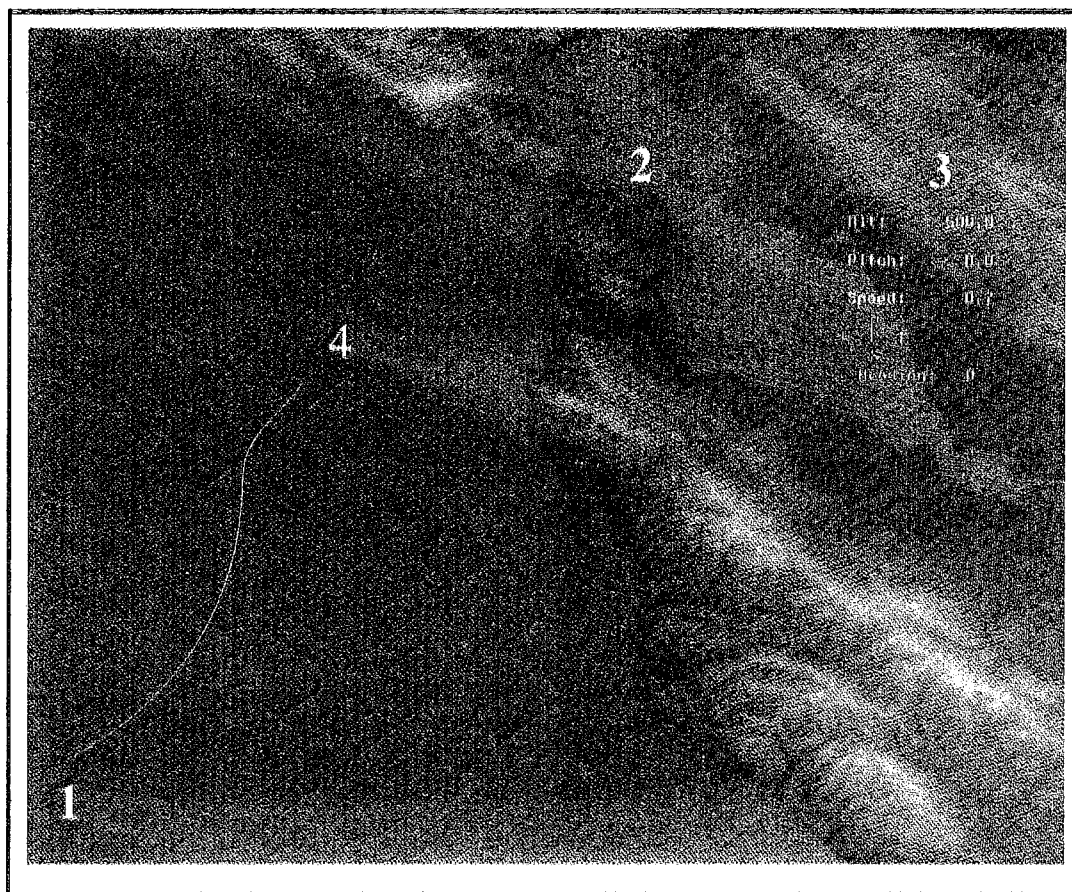
Figure A-4: The AVO Module Display Using The San Francisco Database

The AVO module display shows the two-dimensional representation of the San Francisco area. The AVO module begins with the UAV starting from the lower left hand corner of the display as indicated by the number one in Figure A-4. The objective in this simulation is to confirm the presence of a Soviet produced ZSU 23/24 Air Defense Artillery weapon system on Alcatraz island. Alcatraz is located in the vicinity of the number 2 in Figure A-4. The AVO controls the speed, heading, and pitch of the UAV as he maneuvers the UAV against this target. The AVO is able to use either flight control sticks or the workstation's keyboard - whichever is available. Table A-2 shows the keyboard function mappings for using the keyboard to control the UAV. As the UAV moves from location 1 to its

67

objective at location two the AVO display produces a "snail trail" which indicates the path taken by the UAV from its starting position to its present location. This is illustrated in Figure A-4 by the red line running from location 1 to location 4. Along the way, the heads up display on the AVO position provides the operator with feedback on the speed, altitude, and heading of the UAV. The HUD is located in the vicinity of the number three in Figure A-4. Every.05 seconds, the AVO module sends an update over the network to the MPO position which renders the three-dimensional view of the simulation. In the event that the AVO either moves too close to the ZSU 23/24 located on Alcatraz Island or the operator crashes the UAV, the simulation will start over. The previous snail trail will remain intact and a new mission will begin using another color for the snail trail.

Keypad Key	Function
+	increase speed
-	decrease speed
8	increase pitch
2	decrease pitch
6	change heading to the right
4	change heading to the left
9	roll to the right
3	roll to the left
ESCAPE	Exit Program

TABLE A-1: AVO Module Keyboard And Function Mappings

As the AVO module sends updates over the network, the MPO module listens, parses the protocol data units and renders the three-dimensional view of the UAV. Figure A-5 shows the MPO display as the UAV approaches Alcatraz Island. The MPO module shows

the terrain representation based on the position of the UAV as communicated by the AVO module and the orientation of the UAV's payload as controlled by the MPO operator. The control panel, located at the bottom of the display, provides feedback of key UAV and payload parameters. From left to right the control panel consists of a fuel gauge, a speedometer, and an altimeter. Next, the control panel depicts the heading of the UAV using a compass. The compass rotates to show the heading of the UAV while the red line on the compass indicates the orientation of the UAV's payload in relation to the direction of flight. The next instrument, the artificial horizon, shows the UAV's attitude at that point in the mission. The next two instruments provide feedback on the payload. The Field of View (FOV) gauge, simulates the operation of the payload's zoom lens. Lastly, the instrument on the far right, provides the user information concerning the depression angle of the UAV's payload. There is an additional heads up display (HUD) on the MPO screen that replicates the actual HUD of the UAV. This information includes the lattitude, longitude, range to target, time, and depression angle of the payload.

As is the case with the AVO module, the MPO module can be controlled using either flight control sticks or the workstation's keyboard. When using flight control sticks, the throttle is used to slew the camera left and right. By pushing forward and pulling back on the joy stick, the user can adjust the depression angle of the payload. When using control sticks, the user exits the simulation by pressing the stick's middle button. In the event that the user is using the keyboard, the keyboard function mappings are shown in the table below. Since the simulator's modules are implemented using multiple processors, some processed may still be running upon the termination of the program. There is a shell script called slay which is in both the AVO and MPO directories, enter the slay command to ensure that there are no processes related to the simulator running in the background.
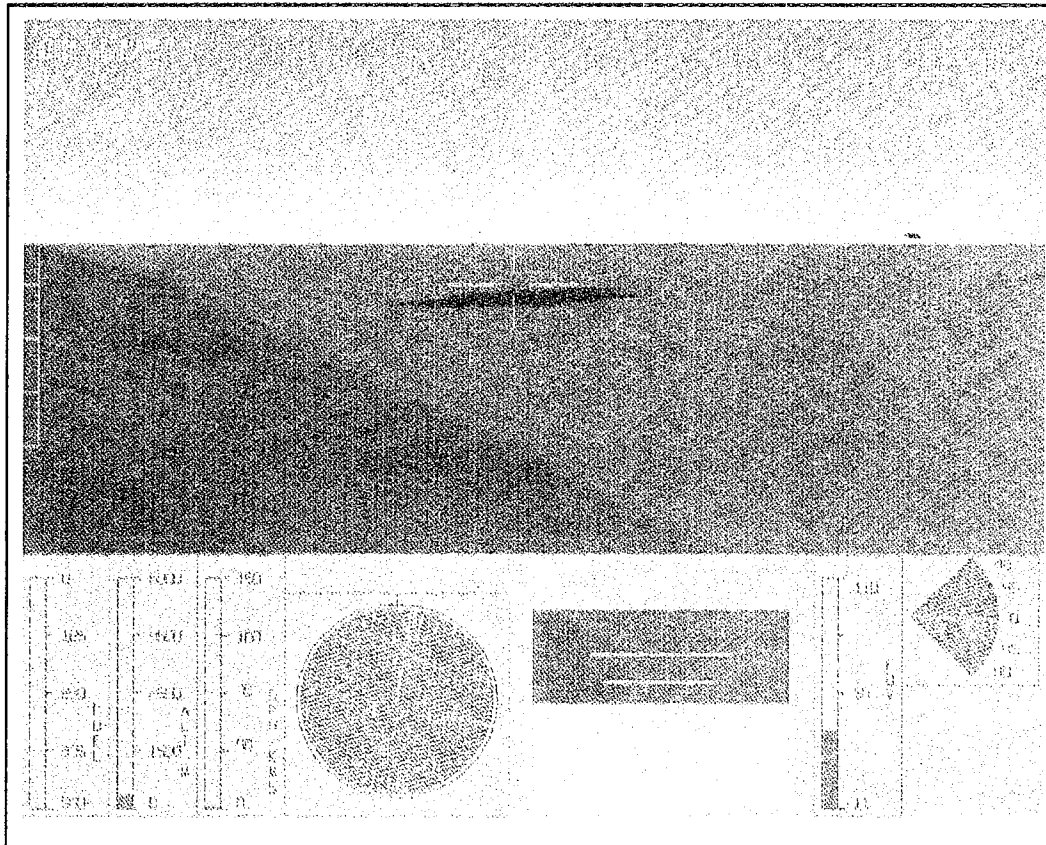
Figure A-5: The MPO Module Display Using The San Francisco Database

Keyboard Key	Function
+	decrease camera field of view
-	increase camera field of view
6	slew camera left
4	slew camera right

TABLE A-2: MPO Module Keyboard And Function Mappings

Key-board Key	Function
8	increase depression angle
2	decrease depression angle
ESCAPE	quit program

TABLE A-2: MPO Module Keyboard And Function Mappings

LIST OF REFERENCES

[BRIE92]   Briefing notes furnished by J. Vaughn, Colonel, United States Army, *Intelligence Support to Future Warfare*, United States Army Intelligence Center and School, Fort Huachuca, Arizona, 27 October 1992.

[CASN93]   Casner, S., "Frequently Asked Questions (FAQ) on the Multicast Backbone," *file://venera.isi.edu/mbone/faq.txt*, May 6, 1993.

[DEFE86]   Defense Mapping Agency, *Defense Mapping Agency Product Specifications for Digital Terrain Elevation Data (DTED)*, DMA Aerospace Center, April 1986.

[DIGI93]   Digital Equipment Corporation, *Digital's Solution to High-Speed Network Computing: FDDI EISA NICs*, White Paper, 1993.

[EVAN94]   Evans, D., Howard, W., "Technology for the Digital Battlefield," *Army Research, Development and Acquisition Bulletin*, p 36-37, July-August 1994.

[FOGG94]   Fogg, Chad, "MPEG-2 source code and MS-DOS Executables Available via Anonymous FTP", file://ftp-request@netcom.com/cfogg/mpeg2, May 17, 1994.

[GUGL93]   Guglielmo, C., "Lossy Compression Standards Have Extra Squeeze Appeal," *Mac Week*, v.7, pp 30-36, 14 June 1993.

[GAIL93]   Gailly, Jean-loup, "Frequently Asked Questions (FAQ) on Computer Compression", July 11 1993. *file://rtfm.mit.edu/compression-faq/part2.*

[HART94]   Hartman, J., Creek, P., *IRIS Performer Programming Guide*, Silicon Graphics Inc., 1994.

[LEHT94]   Lehtinen, R., "What's In Store," *AV Video*, V.16, pp 23-28, 19 May 1994.

[MZPB94]   Macedonia, M.R., Zyda, M.J., Pratt, D.R., Barham, P.J., Zeswitz, S., "NPSNET: A Network Software Architecture for Large Scale Virtual Environments," *Presence*, v.3, Fall 1994.

[INGL93]   Inglis, A., *Video Engineering*, McGraw-Hill, 1993.

[JANE93]   Jane's, *Battlefield Surveillance Systems 1993-94*, 5th Edition, pp 198-204, Jane's Information Group, Sentinel House, 1993.

[HESL93]   Heslop, B., Angell, D., *Mastering Solaris 2*, Sybex, 1993.

[JAIN91]   Jain, R., *The Art of Computer Systems Performance Ananlysis*, John Wiley & Sons, Inc., 1991.

[JAIN94]   Jain, R., *FDDI Handbook, High-Speed Networking Using Fiber and Other Media*, Addison Wesley, 1994.

[LANT93]   LAN TIMES, *Buyers Directory Issue*, McGraw-Hill, August 1993.

[MINO91]   Minoli, D., *Telecommunications Technology handbook*, Artech House, Inc., 1991.

[NNBM94]   Neal Nelson Bench Mark, *Business Benchmark Test Descriptions*, Neal Nelson & Associates, Chicago IL, 1994.

[NPI93]   Network Peripherals Inc., *SBus FDDI Interface, User's Manual*, Netwok Peripherals Inc., 1993.

[PART94]   Partridge, C., *Gigabit Networking*, Addison-Wesley Publishing Co., 1994.

[SASI91]   Littell, R., Freund, R., Spector, P., *SAS System for Linear Models*, SAS Institute Inc, 1991.

[SGIF94]   SGI-faq@viz, SGI Performer Frequently Asked Questions, *file://rtfm.mit.edu:/pub/usenet/comp.sys.sgi.misc*, January 6, 1994.

[STAL91]   Stallings, W., *Data and Computer Communications*, 3rd ed., Macmillan Publishing Co., 1991.

[STEV94]   Stevens, R., *TCP/IP Illustrated, Volume 1, The Protocols*, Addison Wesley, 1994.

[ZESW93]   Zeswitz, S., *NPSNET: Integration of Distributed Interactive Simulation (DIS) Protocol For Communication Architecture and Information Exchange*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1993.

INITIAL DISTRIBUTION LIST

Defense Technical Information Center   2
Cameron Station
Alexandria, VA   22304-6145

Dudley Knox Library   2
Code 052
Naval Postgraduate School
Monterey, CA   93943

Commander   2
Company A, 743rd Military Intelligence Battalion
ATTN: CPT Tipton
Menwith Hill Station
APO AE 08468

Chairman   2
Computer Science Department
Naval Postgraduate School
Monterey, CA   93943

D. R. Pratt, Code CS/Pr   2
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943-5000

Professor Michael J. Zyda, Code CS/Zk   2
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943-5000

Major Michael Macedonia   1
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943-5000

Commander
Attn: ATZS-CDT-I (CPT Earl)   1
USAICS & FH
Fort Huachuca, Arizona 86513-2000

Commanding General
STRICOM   1
12350 Research Parkway
ATTN: Stan Goodman
Orlando, FL 32826-3276